Grid Refinement and Verification Estimates for the RBF construction Method of Lyapunov Functions

by

Najla Abdullah Mohammed

A thesis submitted for the degree of

Doctor of Philosophy

in the

University of Sussex

School of Mathematical and Physical Sciences

July 2016

Declaration

I hereby declare that this thesis has not been and will not be, submitted in whole or in part to another University for the award of any other degree.

Signature:

Acknowledgements

Thank you Allah for always being there for me and for blessing me more than I deserve.

This Thesis would not have been possible unless the help and support of people around me throughout my PhD journey. Above all, I would like to express my sincere gratitude to my Supervisor Dr. Peter Giesl, who supported me with his magnificent guidance, motivation, patience and understanding. I will be forever grateful for his support and for every moment he spent to discuss my work.

It is my pleasure to thank Dr.Siggi for his valuable assistance with C++ programming and for his help and support all the time.

A hearty thanks to my parents Abdullah and Fatimah, for their love and support throughout my life and while I was away. Thank you both for supporting my passion in learning and for giving me strength to achieve the best in my life. Words would never say how grateful I am to both of you.

To my dear husband Hussain, I am truly grateful for sticking by my side, for your unconditional love and care. Thank you for being a great supporter during my good and bad times and for being my family while I am away form them.

A special thanks to my brothers Mohammed and Ahmed and to my sisters Nada, Nahed and shahd, for their support, encouragement and love.

Finally, I would like to express the deepest appreciation to Umm Al-Qura university and to my government ruled by King Salman Bin Abdul-Aziz Al-Saud and previously by King Abdullah Bin Abdul-Aziz Al-Saud (May Allah have mercy on him), for funding my study and for giving me this golden opportunity to continue my higher education abroad.

Grid Refinement and Verification Estimates for the RBF construction Method of Lyapunov Functions

A thesis submitted for the degree of Doctor of Philosophy

University of Sussex

July 2016

Najla Abdullah Mohammed

Abstract

Lyapunov functions are functions with negative orbital derivative, whose existence guarantee the stability of an equilibrium point of an ODE. Moreover, sub-level sets of a Lyapunov function are subsets of the domain of attraction of the equilibrium. In this thesis, we improve an established numerical method to construct Lyapunov functions using the radial basis functions (RBF) collocation method. The RBF collocation method approximates the solution of linear PDE's using scattered collocation points, and one of its applications is the construction of Lyapunov functions. More precisely, we approximate Lyapunov functions, that satisfy equations for their orbital derivative, using the RBF collocation method. Then, it turns out that the RBF approximant itself is a Lyapunov function.

Our main contributions to improve this method are firstly to combine this construction method with a new grid refinement algorithm based on Voronoi diagrams. Starting with a coarse grid and applying the refinement algorithm, we thus manage to reduce the number of collocation points needed to construct Lyapunov functions. Moreover, we design two modified refinement algorithms to deal with the issue of the early termination of the original refinement algorithm without constructing a Lyapunov function. These algorithms uses cluster centres to place points where the Voronoi vertices failed to do so.

Secondly, we derive two verification estimates, in terms of the first and second derivatives of the orbital derivative, to verify if the constructed function, with either a regular grid of collocation points or with one of the refinement algorithms, is a Lyapunov func-

tion, i.e., has negative orbital derivative over a given compact set. Finally, the methods are applied to several numerical examples up to 3 dimensions.

Contents

List of Figures

List of Symbols

$'$	orbital derivative $V'(x) = \langle \nabla V(x), f(x) \rangle$, cf. Definition 1.6.
$.$	temporal derivative $\dot{x}(t) = \frac{d}{dt}x(t)$.
$\|.\|$	Euclidean vector norm unless otherwise stated.
$\|.\|_{\max}$	$\|A\|_{\max} = \max\limits_{1 \leq i,j \leq d} \|a_{ij}\|$, where $A \in \mathbb{R}^{d \times d}$.
$\|.\|_2$	$\|A\|_2 = \left(\sum_{i=1}^{d} \sum_{j=1}^{d} (a_{ij})^2 \right)^{\frac{1}{2}}$, where $A \in \mathbb{R}^{d \times d}$.
$A(x_0)$	domain of attraction of x_0, cf. Definition 1.5.
$\delta_{\tilde{x}}$	Dirac's delta distribution at $\tilde{x} \in \mathbb{R}^d$, i.e., $\delta_{\tilde{x}} f(x) = f(\tilde{x})$.
$\mathcal{B}_p$	unit ball of radius 1 under different p-norms.
E_{nh}	small neighbourhood around the equilibrium points, which is excluded from the refinement and checking procedures.
f	$f \in C^{\sigma}(\mathbb{R}^d, \mathbb{R}^d)$, right hand side of the ODE $\dot{x} = f(x)$, cf. (1.1).
$h_{\|.\|_p}$	fill distance under different p-norms.
h_1	density of Y_{od}.
K	compact set $\{x \in \mathbb{R}^d \mid V(x) \leq R\}$, where V is a Lyapunov function, cf. Theorem 1.1.
$co(K)$	the convex hull of the set K.
T	Lyapunov function with $T'(x) = -\bar{c} < 0$ for $x \in A(x_0) \backslash \{x_0\}$, cf. Theorem 1.2.

t	function with approximates T.
Q	Lyapunov function with $Q'(x) = -p(x)$ for $x \in A(x_0)$, cf. Theorem 1.3.
q	function which approximates Q.
x_+	$x_+ = x$ for $x \geq 0$ and $x_+ = 0$ for $x < 0$.
$\Psi_{i,k}$	product functions of an RBF $\psi_i(r)$ and a power function r^k, cf. Remark 5.2.

The examples considered throughout the thesis

First example

The 2-dimensional linear system

$$\begin{cases} \dot{x} = -x, \\ \dot{y} = -y. \end{cases}$$

which will be visited in Section 2.2 (Example 2.1) and Section 3.2.1 (Example 3.1).

Second example

The 2-dimensional non-linear system

$$\begin{cases} \dot{x} = -x - 2y + x^3, \\ \dot{y} = -y + \frac{1}{2}x^2 y + x^3. \end{cases}$$

which will be visited in the following sections:
Section 3.2.1 (Example 3.2), Section 3.3 (Example 3.5), Section 4.2.2 (Example 4.1), Section 4.2.4 (Example 4.3), Section 5.4 (Example 5.1), Section 6.2.1 (Example 6.1), Section 6.2.2 (Example 6.3).

Third example

The 2-dimensional non-linear system

$$\begin{cases} \dot{x} = -x(1 - x^2 - y^2) - y, \\ \dot{y} = -y(1 - x^2 - y^2) + x. \end{cases}$$

which will be visited in the following sections:
Section 3.2.1 (Example 3.3), Section 3.3 (Example 3.6), Section 4.2.2 (Example 4.2), Section 4.2.4 (Example 4.4), Section 5.4 (Example 5.2), Section 6.2.1 (Example 6.2), Section 6.2.2 (Example 6.4).

Fourth example

The 3-dimensional non-linear system

$$\begin{cases} \dot{x} = x(x^2 + y^2 - 1) - y(z^2 + 1), \\ \dot{y} = y(x^2 + y^2 - 1) + x(z^2 + 1), \\ \dot{z} = 10z(z^2 - 1). \end{cases}$$

which will be visited in the following sections:
Section 3.2.1 (Example 3.4), Section 3.3 (Example 3.7), Section 4.2.4 (Example 4.5).

Chapter 1

Introduction

The determination of the domain of attraction of an equilibrium is an important task in the analysis and derivation of dynamical systems, arising in many practical applications. Since it is difficult to determine the exact domain of attraction analytically, researchers have been seeking numerical algorithms to determine subsets of the domain of attraction. Most of these methods for computing domains of attraction are based on Lyapunov functions, which are functions that decrease along trajectories of the dynamical system. Sublevel sets of Lyapunov functions are positively invariant subsets of the domain of attraction. The construction of such Lyapunov functions, however, is very challenging.

In the last decades, several numerical methods to construct Lyapunov functions have been developed, for a review see [21]. These methods include the **SOS** (sums of squares) method, which is applicable for polynomial vector fields, introduced in [51] and available as a MATLAB tool box [50]. It constructs a polynomial Lyapunov function by **semidefinite optimization**.

The **CPA** method constructs a CPA (continuous piecewise affine) Lyapunov function using **linear optimization** [26, 27]. A simplicial complex is fixed and the space of CPA functions which are affine on each simplex is considered. This space can be parameterized by the values on the vertices. The conditions of a Lyapunov function are transformed into a set of finitely many linear inequalities at the vertices, which include error estimates ensuring that the CPA Lyapunov function has negative orbital derivative inside each simplex. These linear inequalities are used as constraints of a linear programming problem, which can be solved by standard methods. While in the original method an arbitrarily

small neighborhood of the equilibrium had to be cut out, a revised method can construct a CPA Lyapunov function also near the equilibrium by using a fan-like triangulation near the equilibrium [19].

A different method deals with **Zubov's equation** and computes a solution of this partial differential equation (PDE) [9]; the corresponding generalized Zubov equation is a Hamilton-Jacobi-Bellmann equation. This equation has a viscosity solution, a type of weak solution that requires less smoothness than the classical one, which can be approximated using standard techniques after regularisation at the equilibrium, for example one can use piecewise affine approximating functions and adaptive grid techniques [24]. For more details about the theory of viscosity solutions we refer to [5].

The **cell mapping approach** [30] or **set oriented methods** [12] divide the phase space into cells and compute the dynamics between these cells; they have also been used to construct Lyapunov functions [25].

The **RBF** (Radial Basis Function) method, a special case of mesh-free collocation, considers a particular Lyapunov function, satisfying a linear PDE and solves it using **mesh-free collocation** [17, 22]. For this method, a set of scattered grid points is used to find an approximation to the solution of the linear PDE. It is computed by solving a linear system of equations.

In this Thesis, we will present our contributions to improve the RBF construction method of Lyapunov functions.

1.1 Outline of the Thesis

The rest of **this Chapter** gives the necessary background on Dynamical systems, the characterization of a Lyapunov function, and the existence of Lyapunov functions.

In **Chapter 2**, we introduce the construction method of Lyapunov function using radial basis functions. At first, we will explain in detail the radial basis function collocation method for the numerical solutions of PDEs. Then, a description of the construction method for Lyapunov functions, using the RBF collocation method, is provided along with numerical examples.

The forthcoming Chapters are our own new work.

In **Chapter 3**, we improve the RBF construction method by combining it with a grid refinement algorithm. This algorithm uses the Voronoi vertices to refine the grid. It shows a great advantage by reducing the required number of grid points as well as the time needed to construct a Lyapunov function, in 2-D and 3-D examples. This part was published in [47].

Chapter 4 considers the problem of the early termination of the refinement algorithm without constructing a Lyapunov function. To deal with this issue, we design two modified grid refinement algorithms, which keep refining the grid by using the cluster centres, until a successful construction of a Lyapunov function is achieved.

In **Chapter 5**, we derive two verification estimates for the negativity of the orbital derivative of the functions constructed with the RBF method. This Chapter is devoted to analyse the effects of different factors on the final formulation of the estimates. The factors considered are: the different norms and the different distributions of grid points, namely the square and the body centred square configurations.

Chapter 6 combines the results of the preceding Chapters in one method called *The combination method*. This method applies the verification estimates to the RBF functions constructed with either a regular grid, the refinement algorithm, or the modified refinement algorithms.

1.2 General Background

1.2.1 Dynamical Systems

Systems motivated by meteorological phenomena, chemical interactions, biological examples and computing systems can be modelled by differential equations. The fact that many differential equations cannot be solved analytically has encouraged many researchers to come up with different numerical methods for solving such equations.
In our study, we will consider the following autonomous system of differential equations, which defines a dynamical system. Let

$$\dot{x} = f(x) \tag{1.1}$$

with $f \in C^{\sigma}(\mathbb{R}^d, \mathbb{R}^d), \sigma \geq 1, d \in \mathbb{N}$. The initial value problem $\dot{x} = f(x),\ x(0) = \xi$, has a unique solution $x(t)$ which passes through a point $\xi \in \mathbb{R}^d$ at time $t = 0$. The solution $x(t)$ depends continuously on the initial value ξ and is defined for all $t \in I$, where $0 \in I \subset \mathbb{R}$ and I is the maximal interval where the solution exists.

The theory of dynamical systems describes how a system changes over time. The following section gives a brief introduction to dynamical systems including some relevant terminologies and definitions.

In [62], a general definition of a dynamical system was given as an evolution rule that defines a trajectory as a function of a single parameter (time) on a set of states (the phase space). Based on this definition we can say that a dynamical system consists of three components :

- **Complete metric space** (phase space): the set of all states of a given system, $X = \mathbb{R}^d$.
- **Time** ($\mathbb{T}$): which either be discrete ($\mathbb{T} = \mathbb{N}_0$) or continuous ($\mathbb{T} = \mathbb{R}_0^+$).
- A **function** (rule): $S_t x : X \to X$, mapping the system's state at time 0 to time t.

There are two classes of dynamical systems: discrete and continuous. In our study we will only focus on continuous dynamical systems.

Definition 1.1. (Flow Operator for ODE) *Define the operator S_t by $S_t\xi := x(t)$, where $x(t)$ is the solution of the initial value problem $\dot{x} = f(x),\ x(0) = \xi \in \mathbb{R}^d$ for all $t \in \mathbb{R}$, for which this solution exists.*

Assume that the solution exists for all $t \geq 0$, then the flow operator S_t defines a dynamical system.

Definition 1.2. (Continuous dynamical systems) *We call $(X, \mathbb{R}_0^+, S_t)$ a continuous dynamical system, if X is a complete metric space and $S_t : X \to X$ is defined for all $t \in \mathbb{R}_0^+, (t, x) \mapsto S_t x$ is a family of continuous mappings with respect to both x and t. Moreover, we assume that S_t is a semi-group, i.e.*

- *$S_{t+s} = S_t \circ S_s$ for all $t, s \geq 0$.*

- $S_0 = id_X$.

The simplest solutions of the system (1.1) are *equilibrium* solutions, i.e. solutions that do not change in time.

Definition 1.3. (Equilibrium) *A point $x_0 \in \mathbb{R}^d$ is an equilibrium point of (1.1), if $f(x_0) = 0$.*

If x_0 is an equilibrium, then $x(t) = x_0$ is a constant solution for all $t \geq 0$.

Stability of an equilibrium point is a very important property in applications. Studying the behaviour of solutions in a neighbourhood of an equilibrium leads us to the fact that not all equilibria are the same. We call an equilibrium **stable**, if solutions remain near the equilibrium. Moreover, it is called **asymptotically stable**, if solutions remain near the equilibrium and also they converge to it as time tends to infinity.

Definition 1.4. (Stability of an equilibrium) *Let x_0 be an equilibrium. x_0 is called*

1. *Stable : if for all $\epsilon > 0$ there is a $\delta > 0$ such that $\|S_t x - x_0\| < \epsilon$ for all $t \geq 0$ and for all $x \in \mathbb{R}^d$ such that $\|x - x_0\| < \delta$.*

2. *Asymptotically stable : if it is stable and there is a $\delta' > 0$ such that $\|S_t x - x_0\| \overset{t\to\infty}{\longrightarrow} 0$ holds for all $\|x - x_0\| < \delta'$.*

3. *Exponentially asymptotically stable (with exponent $-\nu < 0$) : if it is stable and there is a $\delta' > 0$ such that $\|S_t x - x_0\| \, e^{+\nu t} \overset{t\to\infty}{\longrightarrow} 0$ holds for all $\|x - x_0\| < \delta'$.*

4. *Unstable : if it is not stable.*

One way of analysing the stability of an equilibrium is by using what is called *linear stability analysis*, i.e., an analysis based on the sign of the real parts of the eigenvalues of the system matrix. Consider a linear system of differential equations $\dot{x} = Ax$, where $f(x) = A$ and A is an $d \times d$ matrix. The equilibrium point of the linear system is asymptotically stable if and only if all real parts of the eigenvalues λ of A are negative, i.e. $Re(\lambda) < 0$. If at least one of the eigenvalues has positive real part then the equilibrium is unstable.
For non linear systems, the local behaviour near a hyperbolic equilibrium point x_0 can be determined by the behaviour of the linearised system around x_0, i.e., $\dot{x} = Ax$, where $A = Df(x_0)$ is the Jacobian of f in x_0. An equilibrium point is called hyperbolic if all

of the eigenvalues of the matrix $Df(x_0)$ have non-zero real parts. Then if all real parts of the eigenvalues of $Df(x_0)$ are negative, the equilibrium is asymptotically stable with respect to the linearised system and the original non linear system.

Definition 1.5. (Domain of attraction) *The domain of attraction of an asymptotically stable equilibrium is the region defined by the set of all solutions which converge to x_0, that is*

$$A(x_0) := \left\{ x \in \mathbb{R}^d \mid S_t x \overset{t\to\infty}{\longrightarrow} x_0 \right\} \tag{1.2}$$

One of our main goals is the determination of the domain of attraction of an equilibrium. We can actually compute a subset of the domain of attraction through sub-level sets of a Lyapunov function.

1.2.2 Lyapunov Functions

The method of Lyapunov functions enables us to determine subsets of the domain of attraction of an asymptotically stable equilibrium through sublevel sets. A function $V \in C^1(\mathbb{R}^d, \mathbb{R})$ is called a Lyapunov function for the equilibrium x_0 if it has a local minimum at x_0 and a negative orbital derivative in a neighborhood of x_0.

Definition 1.6 (Orbital derivative)**.** *The orbital derivative of a function $V \in C^1(\mathbb{R}^d, \mathbb{R})$ with respect to (1.1) at a point $x \in \mathbb{R}^d$ is defined by*

$$V'(x) = \langle \nabla V(x), f(x) \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the standard scalar product in $\mathbb{R}^d$.

Remark 1.1. *The orbital derivative is the derivative along solutions: with the chain rule we have*

$$\frac{d}{dt} V(S_t x)\Big|_{t=0} = \langle \nabla V(S_t x), \frac{d}{dt} S_t x \rangle\Big|_{t=0} = \langle \nabla V(x), f(x) \rangle = V'(x). \tag{1.3}$$

The following theorem shows how Lyapunov functions are used to find subsets of the domain of attraction; note that the requirement of the local minimum at x_0 is a consequence of the assumptions. The theorem states that sublevel sets of a Lyapunov function are positively invariant subsets of the domain of attraction, see e.g. [17, Theorem 2.24].

Theorem 1.1. *Let $x_0 \in \mathbb{R}^d$ be an equilibrium of $\dot{x} = f(x)$ with $f \in C^1(\mathbb{R}^d, \mathbb{R}^d)$. Let $V \in C^1(\mathbb{R}^d, \mathbb{R})$ be a Lyapunov function and let $K \subset \mathbb{R}^d$ be a compact neighbourhood of x_0. Furthermore, let*

1. *$K = \{x \in \mathbb{R}^d \mid V(x) \leq R\}$ for an $R \in \mathbb{R}$, i.e., K is a sublevel set of V.*
2. *$V'(x) < 0$ for all $x \in K \setminus \{x_0\}$, i.e., V is decreasing along solutions in $K \setminus \{x_0\}$.*

Then x_0 is asymptotically stable, K is positively invariant and $K \subset A(x_0)$ holds. Moreover, $V(z) > V(x_0)$ holds for all $z \in K \setminus \{x_0\}$.

Existence of Lyapunov functions

Over the last century, the Lyapunov theory of dynamical systems has been the most powerful contribution used for analysing the stability of different types of dynamical systems, which are mathematically modelled by ordinary differential equations. In 1892, Alexander Lyapunov introduced two methods: Lyapunov's indirect method and Lyapunov's direct method.

The direct method of Lyapunov (also called Lyapunov's second method), is a technique that allows us to investigate the stability properties of a given system without solving it explicitly. The idea of this method is to find a function V, that is strictly decreasing along a system's trajectories and these trajectories must eventually converge to the equilibrium point of the system. If such function V exists, then it is called a Lyapunov function and the equilibrium point is asymptotically stable.

However, the difficulty of Lyapunov's direct method lies in finding a suitable function V. This problem gives rise to the converse concept of Lyapunov's direct method; i.e., given that an equilibrium is stable or asymptotically stable, does a suitable Lyapunov function V exist? And if it does exist, how it can be constructed ?

Simply, the converse Lyapunov theorems are results that guarantee the existence of a Lyapunov function under some stability conditions.

In the case of linear systems, Lyapunov answered the converse question via his indirect methods, which says very briefly: if a linear system (I) $\dot{x} = Ax$, $x \in \mathbb{R}^d$ has an asymptotically stable origin, then by [44] there exists a unique positive definite solution $P \in \mathbb{R}^{d \times d}$, satisfying the matrix equation $A^\intercal P + PA = -I_n$, where I_n is the $n \times n$-identity matrix. By

Lyapunov's indirect method, the quadratic function $V(x) = x^\intercal Px$ is a Lyapunov function for (I).

Consequently, a local Lyapunov function can be constructed for non-linear systems by considering the linearisation around the equilibrium point. However, Lyapunov's original work did not guarantee the existence of Lyapunov functions for non-linear systems.

The first converse theorem in the general case was due to Persidskii [52]. Then, in 1949 Massera provided the first converse theorem for asymptotic stability [45]. Many authors have developed Massera's result in several directions to answer the converse questions related to different cases, for an overview see [36] and [19].

Although these converse theorems guarantee the existence of Lyapunov functions, its main drawback is that they do not provide a method to construct Lyapunov functions for non-linear systems explicitly.

In our study, we will assume that x_0 is an exponentially stable equilibrium and we will consider two classes of Lyapunov functions, such that $V'(x) < 0$ holds for all $x \in A(x_0)\backslash\{x_0\}$. We will characterize them by equations for their orbital derivatives $V'(x)$; these are linear first order partial differential equation (PDE) for V. Both functions have the same degree of smoothness as f, i.e. they are C^σ functions.

- The first class are Lyapunov functions T, which satisfy the equation

$$T'(x) = -\bar{c},$$

where $\bar{c} > 0$ is a given constant. Note, however, that the function T is not defined at x_0 and fulfills $\lim_{x\to x_0} T(x) = -\infty$.

To prove the existence of T, we need first to define a non-characteristic hypersurface.

Definition 1.7. *(non-characteristic hypersurface [17, Definition 2.36]). Consider $\dot{x} = f(x)$, where $f \in C^\sigma(\mathbb{R}^d, \mathbb{R}^d)$, $\sigma \geq 1$. Let $h \in C^\sigma(\mathbb{R}^d, \mathbb{R})$. The set $\Omega \subset \mathbb{R}^d$ is called a non-characteristic hypersurface if*

1. *Ω is compact,*
2. *$h(x) = 0$ if and only if $x \in \Omega$,*
3. *$h'(x) < 0$ holds for all $x \in \Omega$, and*
4. *for each $x \in A(x_0)\backslash\{x_0\}$ there is a time $\theta(x) \in \mathbb{R}$ such that $S_{\theta(x)}x \in \Omega$.*

Theorem 1.2. *(Existence of T [17, Theorem 2.38]). Let $\dot{x} = f(x)$, $f \in C^\sigma(\mathbb{R}^d, \mathbb{R}^d)$, $\sigma \geq 1$. Let x_0 be an equilibrium such that $-\nu < 0$ is the maximal real part of all eigenvalues of $Df(x_0)$.*

Let Ω be a non-characteristic hypersurface. Then there is a function $\theta \in C^\sigma(A(x_0)\backslash\{x_0\}, \mathbb{R})$ satisfying

$$S_t x \in \Omega \Leftrightarrow t = \theta(x).$$

Furthermore, $\theta'(x) = -1$ and $\lim_{x\to x_0} \theta(x) = -\infty$.

For all $\bar{c} \in \mathbb{R}^+$ and all functions $H \in C^\sigma(\Omega, \mathbb{R})$ there is a function $T \in C^\sigma(A(x_0)\backslash\{x_0\}, \mathbb{R})$ satisfying

$$T'(x) = -\bar{c} \text{ for all } x \in A(x_0)\backslash\{x_0\} \text{ and}$$
$$T(x) = H(x) \text{ for all } x \in \Omega.$$

Moreover, $\lim_{x\to x_0} T(x) = -\infty$.

- The second class are Lyapunov functions which satisfy

$$Q'(x) = -\|x - x_0\|^2$$

or a similar right-hand side.

Before stating the existence theorem for this class of Lyapunov functions, we first define positive definite functions through class $\mathcal{K}$ functions.

Definition 1.8. *[20, Definition 2.7].*

1. *A continuous function $\alpha : [0, +\infty[\to [0, +\infty[$ is said to be of class $\mathcal{K}$ if $\alpha(0) = 0$ and α is strictly monotonically increasing.*
2. *Let $\mathcal{U}$ be a neighborhood of the origin, and let $p : \mathcal{U} \to \mathbb{R}$ be a locally Lipschitz continuous function. We say that p is a positive definite function if $p(0) = 0$ and there exists a class $\mathcal{K}$ function α such that $p(x) \geq \alpha(\|x\|_2)$ for all $x \in \mathcal{U}$.*

Theorem 1.3. *(Existence of Q). Let x_0 be an equilibrium of $\dot{x} = f(x)$ with $f \in C^\sigma(\mathbb{R}^d, \mathbb{R}^d)$, $\sigma \geq 1$, such that the maximal real part of all eigenvalues of $Df(x_0)$ is $-\nu < 0$. Let $p \in C^\sigma(\mathbb{R}^d, \mathbb{R})$ be a positive definite function. Then there exists a Lyapunov function $Q \in C^\sigma(A(x_0), \mathbb{R})$ with $Q(x_0) = 0$ such that*

$$Q'(x) = -p(x)$$

holds for all $x \in A(x_0)$. *If* $\sup_{x \in A(x_0)} \|f(x)\| < \infty$, *then*

$$K_R \coloneqq \{x \in A(x_0) \mid Q(x) \leq R\}$$

is a compact set in $\mathbb{R}^d$ *for all* $R \geq 0$.

Theorem 1.3 follows from [20, Theorem 2.8]. For more details and background of Theorems 1.2 and 1.3, see [17, Section 2.3].

Chapter 2

Construction of Lyapunov Functions using RBF

2.1 Numerical solutions for PDEs using the Radial Basis Functions collocation method

Meshless collocation based on Radial Basis Function is an effective tool to solve linear PDE's. It has outstanding properties, such as approximating arbitrarily scattered data in multidimensional space as well as providing high order of accuracy which have made it a preferable method for the numerical solutions of PDEs. For a general introduction to Meshless collocation, in particular Radial Basis Functions, see [8, 61]. For the application of RBF to the construction of Lyapunov functions, see [17], where details for the following overview of the method can be found, as well as [22].

A Radial Basis Function is a real-valued function whose value depends only on the distance from the origin i.e., $\Psi(x) = \psi(\|x\|)$, where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^d$. There is a one-to-one correspondence between a Radial Basis Function, or more generally kernel, and its Reproducing Kernel Hilbert Space (RKHS), see [61]. The approximate solution of the PDE will be a norm-minimal interpolant in the RKHS; in our brief overview, however, we do not discuss this relation further, the interested reader is referred to [22].

In the following section we will introduce a family of compactly supported Radial Basis Functions that enables us to approximate functions with certain smoothness, i.e. not

necessarily C^∞. Note that the corresponding RKHS is a Sobolev space with equivalent norm.

The Wendland functions, introduced by Wendland [60], are compactly supported Radial Basis Functions, which are polynomials on their support.

Definition 2.1 (Wendland Functions)**.** *Let $l \in \mathbb{N}$, $k \in \mathbb{N}_0$. We define by recursion*

$$\psi_{l,0}(r) = (1-r)_+^l \tag{2.1}$$

and

$$\psi_{l,k+1}(r) = \int_r^1 t\psi_{l,k}(t)dt \tag{2.2}$$

for $r \in \mathbb{R}_0^+$. Here we set $x_+ = x$ for $x \geq 0$ and $x_+ = 0$ for $x < 0$.

If we fix the parameter $l := \lfloor \frac{d}{2} \rfloor + k + 1$ depending on the space dimension d and the parameter k, then the function $\Psi(x) = \psi_{l,k}(c\|x\|)$ with $c > 0$ is a C^{2k} function with compact support. For dimensions $d = 2$ or $d = 3$, we give some Wendland functions in the following table.

k	$\psi_{l,k}$
1	$\psi_{3,1}(cr) = (1-cr)_+^4(4cr+1)$
2	$\psi_{4,2}(cr) = (1-cr)_+^6(35c^2r^2+18cr+3)$
3	$\psi_{5,3}(cr) = (1-cr)_+^8(32c^3r^3+25c^2r^2+8cr+1)$
4	$\psi_{6,4}(cr) = (1-cr)_+^{10}(429c^4r^4+450c^3r^3+210c^2r^2+50cr+5)$

Table 2.1: The Wendland functions $\psi_{3,1}(cr)$, $\psi_{4,2}(cr)$, $\psi_{5,3}(cr)$ and $\psi_{6,4}(cr)$ with $l = k+2$ and a scaling parameter $c > 0$. Note that these functions are the Wendland functions of Definition 2.1 up to a constant.

Now let us consider a general linear partial differential equation of the form

$$Lu = g \text{ on } \Omega \subset \mathbb{R}^d, \tag{2.3}$$

where L is a linear differential operator of the form

$$Lu(x) = \sum_{|\alpha| \leq m} c_\alpha(x) D^\alpha u(x). \tag{2.4}$$

In our case, the differential operator L will be given by the orbital derivative of a function V with respect to system (1.1), namely

$$LV(x) := \langle \nabla V(x), f(x) \rangle = \sum_{j=1}^{d} f_j(x) \partial_j V(x) = V'(x) \tag{2.5}$$

The operator L in (2.5) is a first order differential operator of the form (2.4) with $c_{e_j}(x) = f_j(x)$.

Let $X_N = \{x_1, x_2, \ldots, x_N\} \subset \Omega$ be a set of N pairwise distinct points which are no equilibria. Define *Dirac's delta-operator* δ by $\delta_{y_0} g(x) = g(y_0)$. Then we have

$$(\delta_{x_k} \circ L)^x V(x) = LV(x_k) = V'(x_k),$$

where the superscript x denotes the application of the operator with respect to the variable x. The approximant $v : \mathbb{R}^d \to \mathbb{R}$ of V will be given by

$$v(x) = \sum_{k=1}^{N} \beta_k (\delta_{x_k} \circ L)^y \Psi(x - y) \tag{2.6}$$

where $\Psi(x)$ is the Radial Basis Function. The coefficients β_k are determined by claiming that the interpolation condition

$$(\delta_{x_j} \circ L)^x V(x) = (\delta_{x_j} \circ L)^x v(x)$$

is satisfied for all grid points $x_j \in X_N$, or in other words that the PDE is satisfied at all points $x_j \in X_N$. This will lead to a linear system for β

$$\mathbf{A}\beta = \alpha \tag{2.7}$$

If the points x_j are pairwise distinct and no equilibria, then the symmetric matrix $\mathbf{A}$ is positive definite, so in particular non-singular. Hence, the system has a unique solution β. The interpolation matrix entries of $\mathbf{A} = (a_{jk})_{j,k=1,\ldots,N}$ are given by

$$a_{jk} = (\delta_{x_j} \circ L)^x (\delta_{x_k} \circ L)^y \Psi(x - y)$$

and the right-hand side $\alpha = (\alpha_j)_{j=1,\ldots,N}$ is given by

$$\alpha_j = (\delta_{x_j} \circ L)^x V(x) = LV(x_j) = V'(x_j)$$

which are chosen to be one of our Lyapunov functions $V'(x_j) = -\bar{c}$ or $V'(x_j) = -\|x_0 - x_j\|^2$.

Finally, we calculate the approximant $v(x)$ and its orbital derivative $v'(x)$, using the following formulas, by evaluating and taking the orbital derivative of (2.6).

$$v(x) = \sum_{k=1}^{N} \beta_k \langle x_k - x, f(x_k)\rangle \psi_1(\|x - x_k\|), \tag{2.8}$$

$$\begin{aligned} v'(x) = \sum_{k=1}^{N} \beta_k \big[& \psi_2(\|x - x_k\|)\langle x - x_k, f(x)\rangle \langle x_k - x, f(x_k)\rangle \\ & -\psi_1(\|x - x_k\|)\langle f(x), f(x_k)\rangle \big], \end{aligned} \tag{2.9}$$

where ψ_1 and ψ_2 are defined as:

$$\psi_1(r) = \frac{\frac{d}{dr}\psi(r)}{r}, \quad \text{for } r > 0 \tag{2.10}$$

$$\psi_2(r) = \begin{cases} \frac{\frac{d}{dr}\psi_1(r)}{r} & \text{for } r > 0 \\ 0 & \text{for } r = 0 \end{cases} \tag{2.11}$$

Note that ψ_1 can be continuously extended to 0.

Remark 2.1 (The value of $\bar{c}$)**.** *Changing the value of $\bar{c} > 0$ has the effect of multiplying the solution β of the linear system $\boldsymbol{A}\beta = \alpha$ by a positive constant. As a consequence also the value of the approximant v and its orbital derivative v' will be multiplied by the same positive constant. This means that the areas of the phase space, where v' is positive, are independent of the value of $\bar{c}$.*

Indeed, this follows from $\boldsymbol{A}\beta = \alpha = -(1, 1, \ldots, 1)^T\ \bar{c}$, since the interpolation matrix $\boldsymbol{A}$ is independent of the value of $\bar{c}$ and from formulas (2.8) and (2.9).

The following error estimate was given in [22, Corollay 4.11]. Note, that $W_2^\tau(\Omega)$ denotes the usual Sobolev space on $\Omega \subset \mathbb{R}^d$.

Theorem 2.1. *Denote by k the smoothness index of the compactly supported Wendland function and fix $l := \lfloor \frac{d}{2} \rfloor + k + 1$. Let $k > \frac{1}{2}$ if d is odd or $k > 1$ if d is even. Set $\tau = k + (d+1)/2$ and $\sigma = \lceil \tau \rceil$. Consider the dynamical system defined by (1.1), where $f \in C^\sigma(\mathbb{R}^d, \mathbb{R}^d)$. Let x_0 be an exponentially stable equilibrium point of (1.1). Let f be bounded in $A(x_0)$ and denote by $V \in W_2^\tau(A(x_0), \mathbb{R})$ the Lyapunov function satisfying $V'(x) = -\|x - x_0\|^2$.*

The reconstruction v of the Lyapunov function V with respect to the operator (2.5) and a compact set $K \subset \Omega := \{x \in A(x_0) \,|\, V(x) \leq r\}$, $r > 0$, satisfies

$$\|v' - V'\|_{L_\infty(\Omega)} \leq C h^{k-\frac{1}{2}} \|V\|_{W_2^{k+(d+1)/2}(\Omega)} \tag{2.12}$$

where $h := \sup_{x\in\Omega} \min_{x_j\in X_N} \|x - x_j\|$ *denotes the fill distance, i.e. the maximum distance which a point in* Ω *can have from the nearest point in* X_N*.*

Remark 2.2. *The set* K *in the theorem above can be any compact subset of* $A(x_0)$ *as* r *can be chosen so large that the sublevel set* Ω *of* V *satisfies* $\Omega \supset K$*. A similar statement for the function satisfying* $V'(x) = -\bar{c}$ *follows also from [22, Corollay 4.11], but with no data sites on the boundary. Note, however, that in this case the function satisfying* $V'(x) = -\bar{c}$ *is only unique up to a constant, the error estimates on the orbital derivative, however, still hold.*

The error estimate (2.12) implies that the approximant v of the Lyapunov function V is actually a Lyapunov function, i.e. satisfies $v'(x) < 0$, if the grid is dense enough. Let us make this more precise: by choosing the fill distance h so small that $Ch^{k-\frac{1}{2}}\|V\|_{W_2^{k+(d+1)/2}(\Omega)} \leq \epsilon$ for a given $\epsilon > 0$,

1. for $V'(x) = -\|x - x_0\|^2$ we have with $|V'(x) - v'(x)| \leq \epsilon$, hence

$$v'(x) \leq V'(x) + \epsilon = -\|x - x_0\|^2 + \epsilon < 0 \text{ if } \|x - x_0\|^2 > \epsilon. \tag{2.13}$$

2. for $V'(x) = -\bar{c}$ we have with $|V'(x) - v'(x)| \leq \epsilon$, hence

$$v'(x) \leq V'(x) + \epsilon = -\bar{c} + \epsilon < 0, \text{ if } \epsilon < \bar{c}. \tag{2.14}$$

Remark 2.3. *In both cases, the approximation may fail to have negative orbital derivative near the equilibrium* x_0*; in the first case the error estimate requires* $\|x - x_0\|^2 > \epsilon$*, and in the second case the function* V *is not defined in* x_0*. If the equilibrium is exponentially stable, one can use the Lyapunov function of the linearised system, the so-called local Lyapunov function, to deal with this small neighborhood of* x_0*, for details see [17], or a modified method, see [18]. In this thesis, we will not deal with this local problem in more detail.*

The error estimate uses the fill distance as a measure; hence, we are led to choose a uniformly fine grid. In examples, however, it turns out that an approximation with negative orbital derivative can be achieved with fewer points using a non-uniform grid. Moreover, the goal is not necessarily to have a good approximation of V, but to construct a function with negative orbital derivative. For example, when approximating the solution of $V'(x) = -\|x - x_0\|^2$, a larger error is permissible for points far away from the equilibrium.

2.2 Steps of the Construction Method

The construction method is based on considering the Lyapunov functions satisfying the PDEs stated in Section 1.2.2. We will explain this construction method in an example.

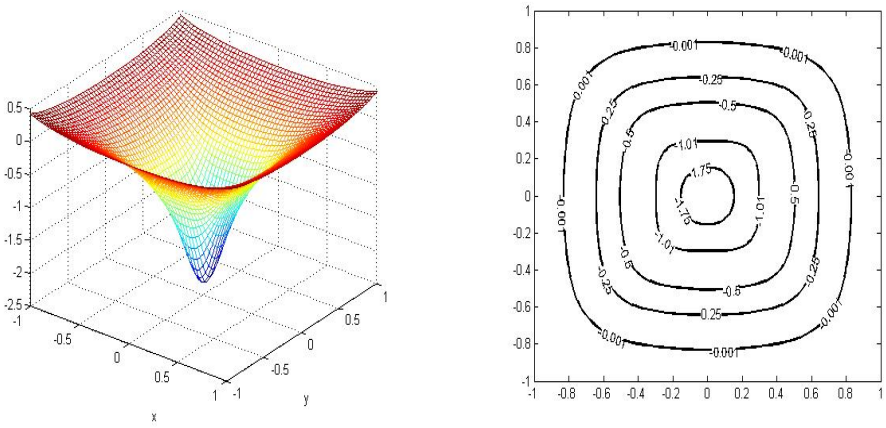
Example 2.1. *Consider the simple linear system*

$$\begin{cases} \dot{x} = -x, \\ \dot{y} = -y. \end{cases}$$

The system has one asymptotically stable equilibrium $x_0 = (0,0)$. Now we will go through the steps of the method.

- *Choose a Radial Basis Function $\Psi(x) = \psi_{l,k}(c\|x\|)$, here we choose the Wendland function $\psi_{6,4}$ with $c = 1$.*
- *Choose a grid $X_N = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^2$, containing no equilibrium point. Here, we fix a regular grid $X_N = \{(x,y) \in \mathbb{R}^2 \mid x, y \in \{0, \pm h, \pm 2h, \ldots, \pm 1\}\} \setminus \{(0,0)\}$, where $h > 0$ is the distance between points in x- and y-direction and will be specified below.*
- *Use the RBF method to approximate the Lyapunov function $V = T$ which satisfies $T'(x) = -1$ by the approximant v and then calculate its orbital derivatives v' using (2.8) and (2.9).*
- *From the ansatz of the Lyapunov function V, we know that the orbital derivative is negative at every point in our grid, i.e., $v'(x_i) < 0$ for all $x_i \in X_N \subset \mathbb{R}^2$, but there may be points in $[-1,1]^2$, where the orbital derivative is positive. The error estimate tells us that if h is small enough, the orbital derivative will be negative except for a small neighborhood of the equilibrium. Hence, we start with a certain h, and check the sign of the orbital derivative at the points between the grid points. If we have points where $v'(x) > 0$, we need a finer grid. Therefore, we choose h smaller and smaller until we have $v'(x) < 0$ for all points in the desired area. In this example, we used $h = 1/6$, resulting in $N = 168$ points.*
- *Find a sublevel set K of the Lyapunov function (the approximant) v of level $R \in \mathbb{R}$ which is a subset of $\{x \in \mathbb{R}^2 \setminus \{(0,0)\} \mid v'(x) < 0\}$; then K is a subset of the domain*

of attraction $A(x_0)$. Figures 2.1 and 2.2 show the functions v and v' as well as level sets of v.



(a) The constructed Lyapunov function $v(x, y)$.

(b) Level sets of $v(x, y)$.

Figure 2.1: (a) shows the constructed Lyapunov function v and (b) different sublevel sets of v for different values of R, which are subsets of the domain of attraction.

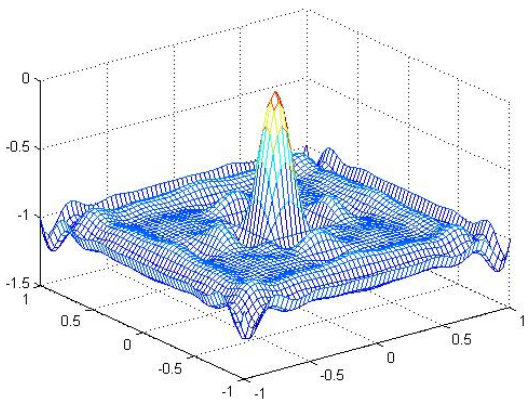


Figure 2.2: The orbital derivative $v'(x, y)$ of the constructed Lyapunov function v; note that $v'(x) \approx -1$, except for a small neighborhood of the origin.

During the calculations of the Lyapunov function v we may find some points with positive orbital derivatives. This occurs because either the grid we have chosen is not fine enough, or the grid points do not all lie in the domain of attraction. We cannot exclude

the second case, so we use a finer grid to calculate the Lyapunov function v and see if the problem will be solved. However, instead of using a regular, finer grid, the question is whether the refinement can be done more efficiently by using an irregular grid with fewer points. Our goal is to have a fine grid but to avoid expensive computations which are caused by refining the whole area rather than only the parts where we need to add more points. Therefore, we combine this construction method with a refinement algorithm.

Chapter 3

Grid Refinement Algorithm

For the numerical solution of partial differential equations (PDE) using mesh-free methods, adaptive refinement techniques play an important role in achieving better accuracy with the minimum number of points. This will be done by only refining (adding more points to) the areas where the solution of the PDE has rapid variations [33, 54, 7, 2]. Since the mesh-free methods, including the RBF approximation method, do not require special connectivity between points, the procedure of adding (or removing) points becomes easy and convenient.

During the last two decades, scientists in the field of scientific computation and engineering have paid a considerable attention to the development of adaptive algorithms for mesh-free methods [13, 39, 56, 43, 15, 3, 42, 49]. Some of the node refinement strategies in the literature include: applying the adaptive mesh refinement (AMR) algorithm to place overlapping refined grids recursively over the regions specified by an error estimator. the refinement procedure stops after the spatial discretization error of the radiative transport equation (RTE) has reached a sufficient level [34]. On the other hand, [38] used a phase indicator to refine nodes in the liquid phase, (by adding four symmetric nodes around the refined node), which needs higher node distribution density comparing to the solid one in solving thermo fluid problems with phase change [38]. Moreover, a local refinement using a local Delaunay triangulation algorithm and other adaptive techniques were presented in [41].

For the RBF construction method, introduced in the previous Chapter, we will combine it with a grid refinement algorithm, aiming for a successful construction of Lyapunov

functions with fewer collocation points and less computation time than the original method

3.1 The Refinement Algorithm

Our proposed algorithm is recursive and uses *Voronoi diagrams*. In each step, given a grid, we generate a Voronoi diagram for our grid points, then we consider the Voronoi vertices of each cell of this diagram as possible points to be added to the grid. Finally, we run a test on each Voronoi vertex and decide whether we add the point to the grid or not.

We have used Voronoi vertices as new possible points for our grid since they are equidistant to three or more previous grid points, and thus lie "in between" the previous grid points. Consequently, we guarantee not to add points that are too close to each other, which would lead to a singular interpolation matrix $\mathbf{A}$ of the linear system (2.7).

We start the first section of this Chapter by a brief introduction to the Voronoi diagrams and its dual structure, the Delaunay triangulation, then describe the strategy of the refinement algorithm. In the second section, we discuss the issue of the unsuccessful termination of the refinement algorithm. The contents of this Chapter were published in [47].

3.1.1 Voronoi Diagram and Delaunay Triangulation

Voronoi diagrams and Delaunay triangulations have enormous applications in different scientific fields, especially in mesh generation and nodes insertion procedures. Sibson [58] developed an interpolation method based on Voronoi diagrams, the method is known as natural neighbour interpolation method. Moreover, there is a refinement algorithm called Ruppert's Delaunay refinement algorithm [57]. Some recent works include [35], presenting a refinement procedure to develop the gradient smoothing method using Delaunay triangulation for the adaptive analysis in solid mechanics, and [64], where a Voronoi neighbour criterion is used to construct the adaptive radial point interpolation method. Voronoi diagrams have also been used in kernel-based adaptive particle methods for numerical flow simulation [31], and for a thinning algorithm in multistep interpolation with Radial Basis Functions [14].

Voronoi Diagram:

A Voronoi diagram is a geometric structure that divides a d-dimensional space into cells based on the distance between sets of points in the space [55]. Many algorithms have been proposed for computing Voronoi diagrams. The fundamental and most popular ones include: The Divide and Conquer algorithm and Fortunes's Sweep Line algorithm [6]. For the purpose of explaining the structure of Voronoi diagrams, we will explain a very simple but less efficient algorithm using perpendicular hyperplanes.

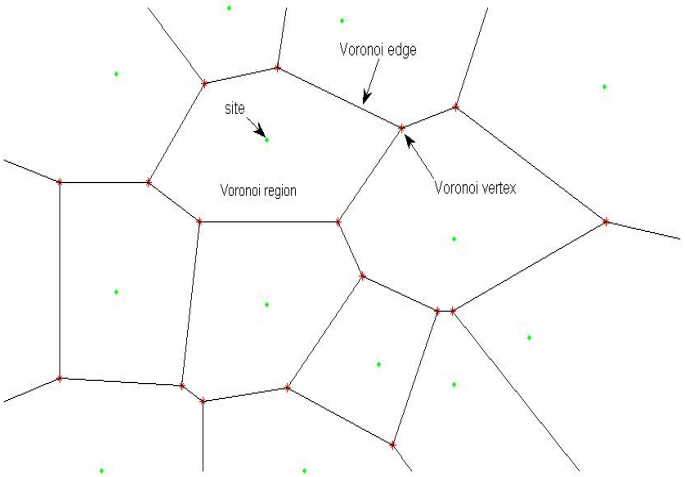


Figure 3.1: The Voronoi diagram for a set of sites (green o) with a Voronoi region, Voronoi edge and Voronoi vertex (red *).

Let $S = \{s_1, s_2, \ldots, s_n\} \subset \mathbb{R}^d$ be a set of n arbitrarily distributed and distinct sites (points) in $\mathbb{R}^d$. The perpendicular bisector algorithm works as follows: for each pair of sites in S we construct a perpendicular hyperplane to the line segment joining these sites. At the end of this process, we will have intersections of finitely many hyperplanes which build up cells, with a convex polygon structure, known as *Voronoi regions*. The boundaries of each region are called *Voronoi edges* and the intersections of Voronoi edges are called *Voronoi vertices*. For more details see [6, 37, 32].

Mathematically, the Voronoi region of a point s_i in S is defined by

$$\mathcal{V}_i = \bigcap_{j=1, j\neq i}^{n} \Big\{ x \in \mathbb{R}^d | \|x - s_i\| < \|x - s_j\| \Big\},$$

where $\|\cdot\|$ denotes the Euclidean distance. This means that for every point $x \in \mathbb{R}^d$ within

a Voronoi region $\mathcal{V}_i$ the Euclidean distance of x to the site s_i, which is also inside the region, is smaller than the Euclidean distance of x to any other site s_j.

Remark 3.1. *As the Voronoi region is the intersection of finitely many hyperplanes, each Voronoi region is a convex polygon.*

Delaunay Triangulation:

Delaunay triangulation is the dual structure of a Voronoi diagram. The relation between them is simply that the sites of the Voronoi diagram are the vertices of the Delaunay triangulation and vice versa. See figure 3.2.
It is defined as a partition of the convex hull of S into $O(n^{\lceil \frac{d}{2} \rceil})$ d-simplices in which the circumsphere, *the sphere passing through the vertices of a simplex*, of each d-simplex is empty, i.e., there are no sites of S in its interior. The Delaunay triangulation is uniquely defined, if all vertices are in a so called general position, i.e., no more than $d+1$ vertices lie on a common sphere [10, Theorem 2.11].

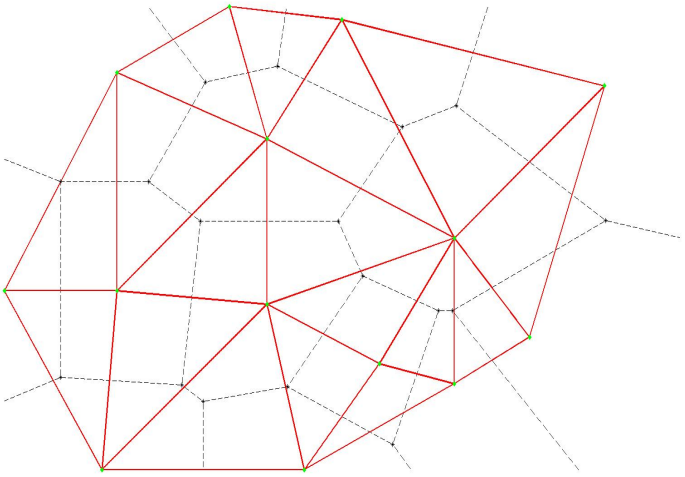


Figure 3.2: The Delaunay triangulation of the Voronoi diagram in figure (3.1). The sites of the Voronoi diagram (green o) are the vertices of the Delaunay triangulation.

3.1.2 The Algorithm Strategy

The first step of the refinement process starts by constructing an RBF approximant v_1 with a coarse grid of collocation points X_1. The Voronoi diagram for X_1 will place the Voronoi vertices in between the grid points in X_1, making them potential points to be added to the grid. Therefore, we check the sign of the orbital derivative v_1' at each Voronoi vertex. Then, we add all the vertices with positive orbital derivative to X_1 and ignore the ones with negative orbital derivative. By the end of this step, we are obtaining a new set of grid points $X_2 = X_1 \cup \{\text{Voronoi vertices with positive orbital derivative}\}$, as well as a new approximant v_2 for X_2.
The algorithm will be repeating this step until it terminates as no new points are added. The implementation of the refinement algorithm follows the following steps.

1. Fix a compact neighbourhood $K \subset \mathbb{R}^d$ of the equilibrium x_0 and a Radial Basis Function. Let $n = 1$ and start with an initial set of grid points $X_1 = \{x_1^{(1)}, x_2^{(1)}, \ldots, x_{N_1}^{(1)}\} \subset K$, not containing any equilibrium.

2. Calculate a Lyapunov function v_n using the RBF method with the grid $X_n = \{x_1^{(n)}, x_2^{(n)}, \ldots, x_{N_n}^{(n)}\}$. In the following steps, we drop the superindex (n) if it is clear in which step we are.

3. Generate Voronoi vertices, $Y_n = \{y_1, y_2, \ldots, y_{M_n}\} \subset \mathbb{R}^d$, for the grid points X_n. Exclude points in Y_n which are equilibria, or which lie in a small neighbourhood E_{nh} of an equilibrium or which lie outside K.

4. Run a test on each vertex in the set Y_n and check whether $v_n'(y_j) < 0 \;\; (y_j \in Y_n^-)$ or $v_n'(y_j) \geq 0 \;\; (y_j \in Y_n^+)$, where $j = 1, \ldots, M_n$ and $Y_n = Y_n^- \cup Y_n^+$.

5. Define new grid $X_{n+1} = X_n \cup Y_n^+$.

6. $n \to n + 1$, repeat the steps 2. to 5. until $Y_n^+ = \emptyset$.

3.2 Numerical Examples

The application of the refinement algorithm on numerical examples has a general structure as follows: the starting grid of the refinement process is a coarse and equidistant grid of

collocation points with fill distance h. Then, we run the refinement algorithm until we successfully construct a Lyapunov function v. After that, we fix a checking grid X_{check} of size h_{check}, which is smaller than h, to check the negativity of the orbital derivative of v. However, we will provide a rigorous estimate for checking the sign of v' in Chapter 5.
It is important to mention that usually there will be a small neighbourhood E_{nh} around the equilibrium point, which is excluded from the refinement and the checking procedures as we cannot expect v' to be negative here, see Remark 2.3.

3.2.1 Two-dimensional Examples

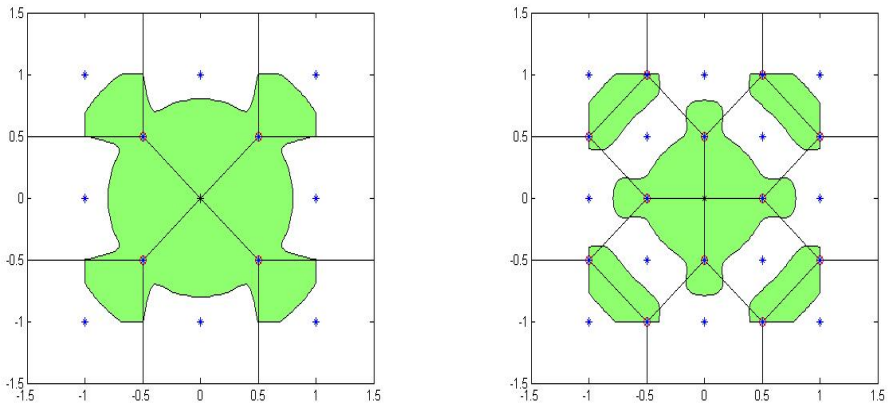
Example 3.1. *Consider the system from Example 2.1. We will explain the refinement algorithm starting with a coarse equidistant grid* $X_1 = \{(x, y) \in \mathbb{R}^2 \mid x, y \in \{0, \pm h, \dots, \pm 1\}\} \setminus E_{nh}$, *where* $E_{nh} = \{(0, 0)\}$ *and* $h = 1$, *i.e.* $N_1 = 8$ *points.*

- *The first step of the algorithm* $n = 1$*: We start with an initial grid* $X_1 = \{x_i\}_{i=1}^{8}$ *and calculate the Lyapunov function* v_1 *on the grid* X_1. *Figure 3.3 (a) shows the level set* $v_1'(x, y) = 0$ *with the area where* $v_1'(x, y) > 0$ *in green. We can also see the Voronoi diagram for our grid points and the circled Voronoi vertices which will all be added to the set* X_1, *since they are all located in the green area. In this case, we have* $Y_1^+ = Y_1$ *and* $Y_1^- = \emptyset$, *i.e. all four new points (the equilibrium is not part of* Y_1*) have positive orbital derivative and will be added to the grid.*

- *The second step of the algorithm* $n = 2$*: Our new set of grid points is now* $X_2 = X_1 \cup \{y_j\}_{j=1}^{4}$. *Figure 3.3 (b) shows the level set* $v_2'(x, y) = 0$. *Again, we generate a Voronoi diagram for the set* X_2 *and determine the Voronoi vertices to be added to the existing grid points, see Figure 3.3 (b).*

- *The third step of the algorithm* $n = 3$*: The new set of grid points is* $X_3 = X_2 \cup \{y_j\}_{j=1}^{12}$. *After going through the same steps again we show in Figure 3.4 (a) the grid and level set* $v_3'(x, y) = 0$.

- *After the fourth step, the set* $Y_4^+ = \emptyset$ *and the algorithm terminates as all Voronoi vertices have negative orbital derivative. Figure 3.4 (b) shows the final set of grid points* $X_4 = X_3 \cup \{y_j\}_{j=1}^{12}$ *and no more areas with positive orbital derivative (green*



(a) The first step $n = 1$ with 8 grid points.

(b) The second step $n = 2$ with 12 grid points.

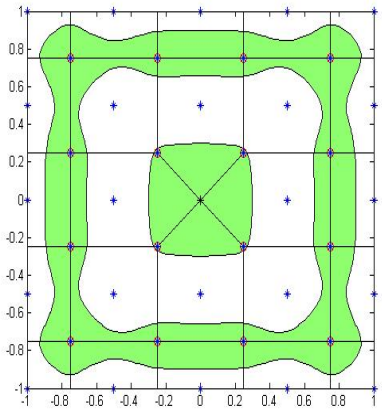
Figure 3.3: The first two steps, $n = 1, 2$, of the refinement algorithm. Both figures show the level set $v'(x, y) = 0$, which divides the region into areas with $v'(x, y) > 0$ (green), and areas with $v'(x, y) < 0$ (white). The grid points are blue * and the Voronoi diagrams with Voronoi vertices (red o) are shown.

> *areas). Thus, we have found a Lyapunov function and will now be able to determine sublevel sets, which are subsets of the domain of attraction. We plot the graph of the Lyapunov function in Figure 3.5 (a) and its sublevel sets in Figure 3.5 (b).*

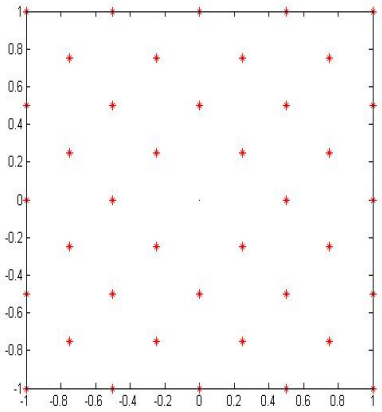
To show that v' is negative, we have checked that the sign of $v'(x, y)$ is negative on the grid $X_{check} = \{(x, y) \mid x, y \in \{0, \pm h_{check}, \pm 2\, h_{check}, \ldots, \pm 1\}\} \setminus \{(0, 0)\}$ with $h_{check} = 10^{-3}$. Note that, in Chapter 5, we will provide a more reliable method to check the negativity of the orbital derivative over a compact set K.

When solving the same example with a regular grid in Example 2.1, we needed $N = 168$ points, whereas with the refinement algorithm we have constructed a Lyapunov function with a grid of only $N_4 = 40$ points, hence we have reached our goal of having a dense enough grid with fewer points. The sublevel sets of the Lyapunov function with the refinement algorithm are similar to those that we obtained with the regular grid $N = 168$, see Figures 2.1 (b) and Figure 3.5 (b).

After successfully testing the refinement algorithm on a linear system, we will now examine it on a nonlinear one.

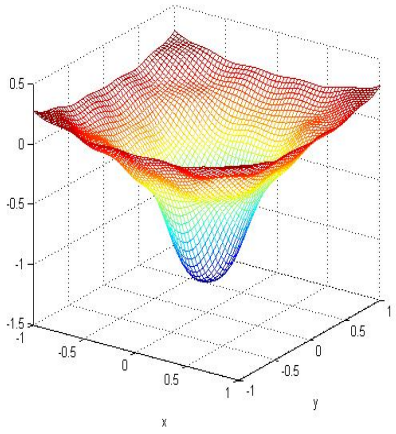


(a) The third step of the refinement algorithm showing the level set $v_3'(x,y) = 0$, in green the areas where $v_3'(x,y) > 0$. The 24 grid points of X_3 (blue *) and Voronoi vertices (red o) to be added to set X_3.

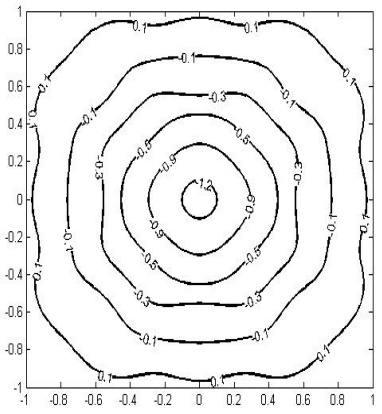


(b) The final set X_4 of 40 grid points (red *) after the refinement algorithm has terminated – no areas where $v_4'(x,y) > 0$ are left.

Figure 3.4: (a) shows the third step of the refinement algorithm, and (b) shows the final set of grid points with no areas of positive orbital derivative.



(a) The constructed Lyapunov function $v_4(x,y)$.



(b) Different sublevel sets of $v_4(x,y)$.

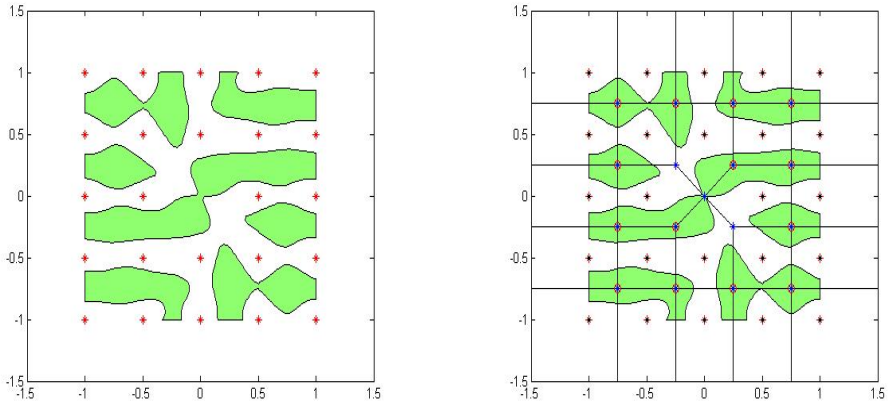
Figure 3.5: (a) The constructed Lyapunov function $v_4(x,y)$ with the refinement algorithm (b) and its sublevel sets for different levels, which are all subsets of the domain of attraction.

Example 3.2. *Consider the non-linear system [22, Example 4.3]*

$$\begin{cases} \dot{x} = -x - 2y + x^3, \\ \dot{y} = -y + \frac{1}{2}x^2 y + x^3. \end{cases}$$

The system has an asymptotically stable equilibrium at $(0, 0)$.

For this example, we approximate the Lyapunov function satisfying $V'(x) = -\|x\|^2$. *We have used the Wendland function* $\psi_{6,4}$ *with c=1 and started with the grid* $X_1 = \{(x, y) \in \mathbb{R}^2 \mid x, y \in \{0, \pm h, \ldots, \pm 1\}\} \setminus E_{nh}$, *where* $E_{nh} = [-0.1, 0.1]^2$ *and* $h = 0.2$, *i.e.* $N_1 = 24$ *points. Figure 3.6 (a) shows the region where* $v_1'(x, y) > 0$ *(green) and Figure 3.6 (b) shows the Voronoi diagram and vertices* Y_1. *The points marked with red o (14 points) have positive orbital derivative and form the set* Y_1^+, *these are the points that will be added to the previous grid. On the other hand, the two blue points (blue *) have negative orbital derivative and form the set* Y_1^-, *thus will not be added to our grid points.*



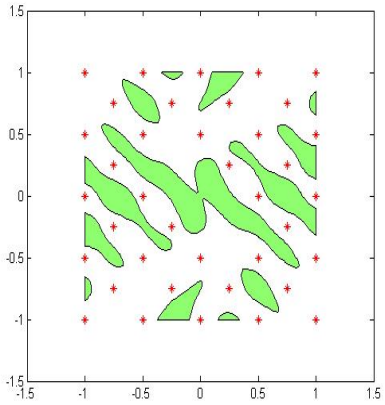
(a) The starting grid with $N_1 = 24$ points.

(b) The Voronoi vertices $Y_1 = Y_1^+ \cup Y_1^-$. The points in the green area (red o) form the set Y_1^+ and will be added to our existing set X_1 of grid points; there are two Voronoi vertices lying in the white set (blue *), they form the set Y_1^- and will not be added to the grid.

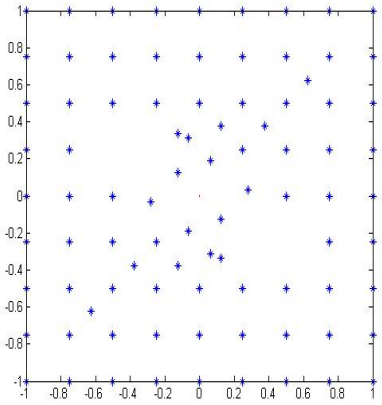
Figure 3.6: Both figures show the level set $v_1'(x, y) = 0$, which divides the region into areas with $v_1'(x, y) > 0$ (green), and areas with $v_1'(x, y) < 0$ (white). The grid points $N_1 = 24$ of the initial grid are red *.

Figure 3.7 (a) shows the level sets of the approximation v_2, using the refined grid $X_2 = X_1 \cup Y_1^+$; the orbital derivative of v_2 for all points of X_2 is negative by construction.

After four refinement steps, the algorithm terminates with 88 grid points and the final function satisfies $v_4'(x, y) < 0$ everywhere; no green areas (where $v_4'(x, y) > 0$) occur in Figure 3.7 (b). To show that v_4' is negative, we have checked that the sign of $v_4'(x, y)$ is negative on the grid $X_{check} = \{(x, y) \mid x, y \in \{0, \pm h_{check}, \pm 2\, h_{check}, \ldots, \pm 1\}\} \setminus E_{nh}$ with $h_{check} = 10^{-3}$.



(a) The level set $v_2'(x, y) = 0$ after the first refinement procedure, recalculated on the new set X_2 of $N_2 = 38$ grid points. Areas where $v_2'(x, y) > 0$ are shown in green.



(b) The grid points $N_4 = 88$ after the termination of the refinement algorithm.

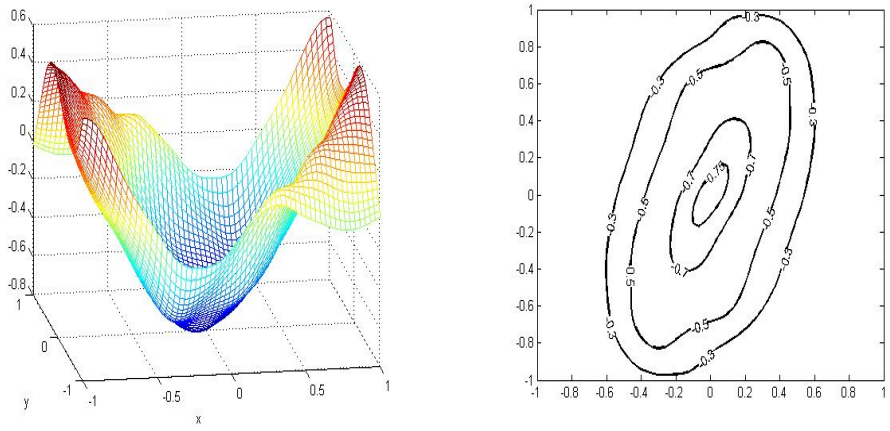
Figure 3.7: (a) The second step of the refinement algorithm and (b) the final step of the refinement algorithm: no areas with positive orbital derivative are left.

Figure 3.8 (a) shows the graph of the constructed Lyapunov function $v_4(x, y)$, and Figure 3.8 (b) shows some of its sublevel sets. To construct a Lyapunov function with a regular grid which has negative orbital derivative on X_{check}, we need a grid of $N = 360$ points.

Example 3.3. *Consider the system [17, Example 2.10]*

$$\begin{cases} \dot{x} = -x(1 - x^2 - y^2) - y, \\ \dot{y} = -y(1 - x^2 - y^2) + x. \end{cases}$$

The system has an exponentially stable equilibrium at $(0, 0)$.



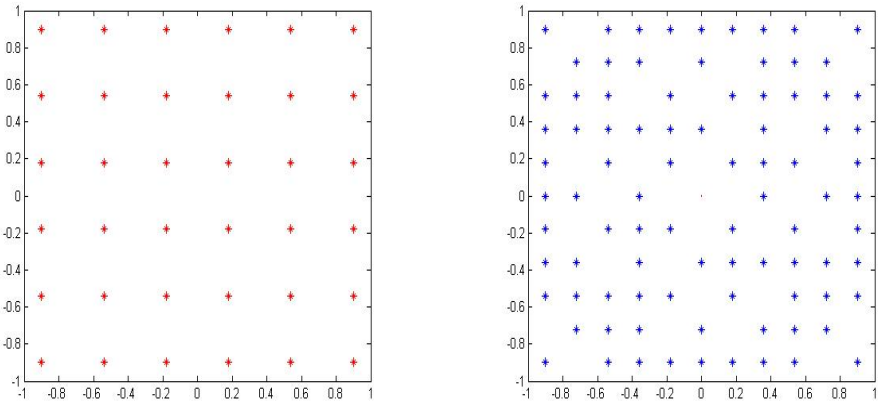
(a) The constructed Lyapunov function $v_4(x, y)$ with the refinement algorithm.

(b) Different sublevel sets of v_4.

Figure 3.8: (a) The constructed Lyapunov function $v_4(x, y)$ with the refinement algorithm and (b) different sublevel sets of v_4.

In this example, we have used the Wendland function $\psi_{6,4}$ with $c = 1$ and approximated the Lyapunov function satisfying $T' = -1$. We start our refinement algorithm with an initial set of regular grid points $X_1 = \{(x, y) \in \mathbb{R}^2 \mid x, y \in \{0, \pm h, \ldots, \pm 0.9\}\} \setminus E_{nh}$, where $E_{nh} = [-0.1, 0.1]^2$ and $h = 0.36$, i.e. $N_1 = 36$ points distributed on a compact set $K = [-0.9, 0.9]^2$, see figure 3.9 (a). After performing four refinement steps, the algorithm successfully constructs a Lyapunov function v_4 with $N_4 = 88$ points, see figure 3.9 (b). The constructed function v_4 satisfies $v_4'(x, y) < 0$ on a checking grid $X_{check} = \{(x, y) \mid x, y \in \{0, \pm h_{check}, \pm 2\, h_{check}, \ldots, \pm 0.9\}\} \setminus E_{nh}$ with $h_{check} = 10^{-3}$. Figure 3.10 (a), shows the Lyapunov function v_4 constructed with the final set of grid points obtained with the refinement algorithm $N_4 = 88$ points. Moreover, figure 3.10 (b), displays different sublevel sets of v_4.

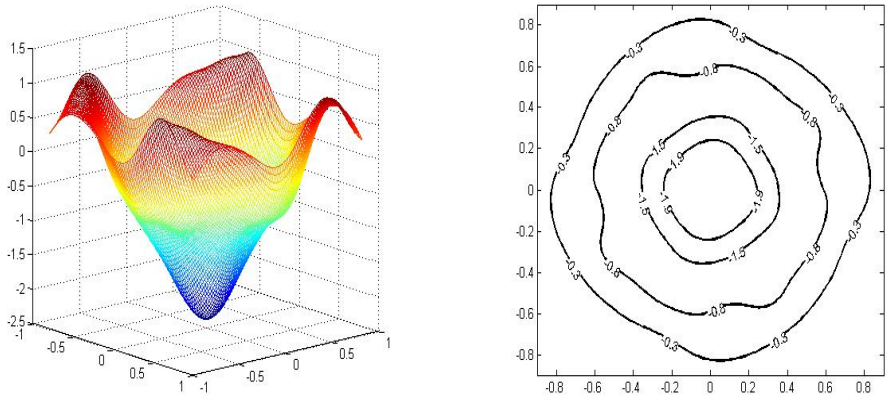
Note that, the domain of attraction of this system is given by $A(0, 0) = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 < 1\}$, so K actually is not a subset of the domain of attraction. As usually the domain of attraction is not known in advance, this is a realistic situation, and even in this situation, the refinement algorithm has worked well.



(a) The initial grid X_1 with $N_1 = 36$ points.

(b) The final grid X_2 with $N_4 = 88$ points.

Figure 3.9: (a) shows the distribution the initial grid points, (b) the distribution of final grid points after the last refinement step.



(a) The constructed Lyapunov function $v_4(x, y)$ with the refinement algorithm.

(b) Different sublevel sets of v_4.

Figure 3.10: (a) The constructed Lyapunov function $v_4(x, y)$ with the refinement algorithm and (b) different sublevel sets of v_4.

3.2.2 Three-dimensional Example

Example 3.4 (Three-dimensional system)**.** *Consider the 3-dimensional system given in [17, Example 6.4]*

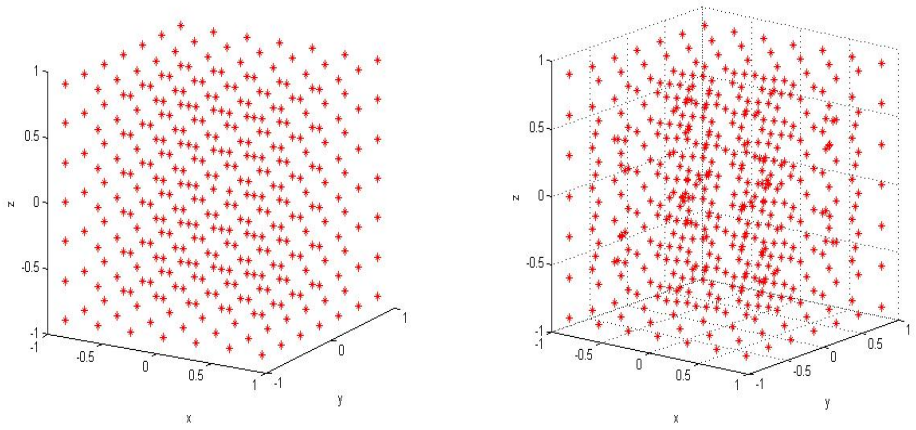
$$\begin{cases} \dot{x} = x(x^2 + y^2 - 1) - y(z^2 + 1), \\ \dot{y} = y(x^2 + y^2 - 1) + x(z^2 + 1), \\ \dot{z} = 10z(z^2 - 1). \end{cases}$$

The system has an exponentially stable equilibrium at $(0, 0, 0)$ *and its domain of attraction is given by*

$$A(0, 0, 0) = \{(x, y, z) \in \mathbb{R}^3 \,|\, x^2 + y^2 < 1, |z| < 1\}.$$

In [17, Example 6.4], an RBF approximation with 137 *points resulted in a large area near the equilibrium with positive orbital derivative; larger than the set* $[-0.2, 0.2]^3$*, which is later excluded in our example. With a modified algorithm, using the Taylor polynomial at the equilibrium, this was overcome in [17, Example 6.4].*

As generally the domain of attraction is not known, we have chosen to use a grid in the set $K = [-0.9, 0.9]^3$*, which is not a subset of the domain of attraction, but also does not include other invariant sets. This is a more realistic, but also more challenging test case for the method.*



(a) The initial grid X_1 with $N_1 = 342$ points.

(b) The final grid X_2 with $N_4 = 458$ points.

Figure 3.11: (a) shows the distribution the initial grid points, (b) the distribution of final grid points after the last refinement step.

We choose the Wendland function $\psi_{6,4}$ *with* $c = 0.6$*. We have started with a regular grid* $X_1 = \{(x, y, z) \in \mathbb{R}^3 \,|\, x, y, z \in \{0, \pm h, \pm 2h, \ldots, \pm 0.9\}\} \backslash E_{nh}$*, where* $E_{nh} = [-0.2, 0.2]^3$

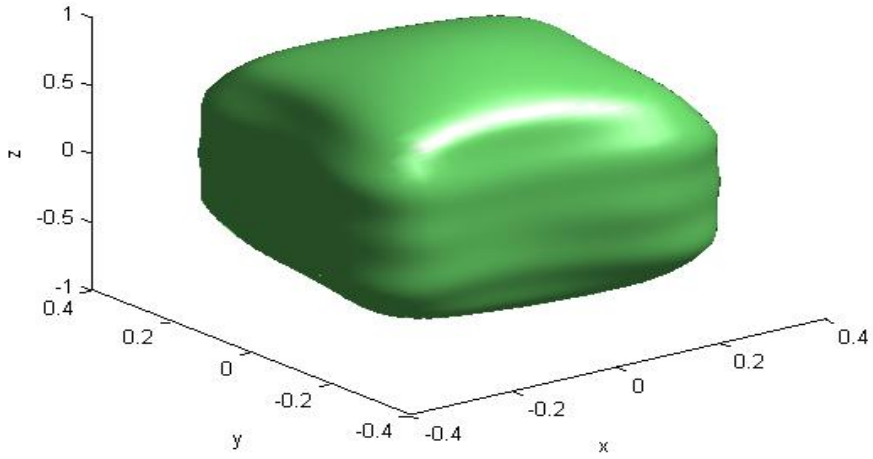


Figure 3.12: The figure shows a level set of the constructed Lyapunov function v_4 at value -1.2474.

and $h = 0.3$, *i.e.* $N_1 = 342$ *points and we have approximated the Lyapunov function satisfying* $V' = -\|x\|^2$. *In order to check the sign of the orbital derivative of the constructed Lyapunov function* v *on finitely many points, we fix a grid* $X_{check} = \{(x, y, z) \in \mathbb{R}^3 \,|\, x, y, z \in \{0, \pm h_{check}, \ldots, \pm 0.9\}\} \backslash E_{nh}$ *with* $h_{check} = 10^{-2}$.

To construct a Lyapunov function v *with a regular grid which has negative orbital derivative on* X_{check} *we need* $N = 2196$ *points* ($h = 0.15$).

Now, applying our refinement algorithm with the initial set $N_1 = 342$ *grid points, gives us a final set of grid points* $N_4 = 458$, *thus reducing again the number of points needed by a factor* 4. *The constructed Lyapunov function* v *has negative orbital derivative on the set* X_{check}. *Figure 3.11 (a) shows the initial grid* X_1, *and Figure 3.11 (b) the grid* X_4 *after the refinement algorithm. Figure 3.12 displays a sublevel set of the Lyapunov function* v_4, *which is a subset of the domain of attraction, obtained at level value* -1.2474.

3.3 Unsuccessful termination of the refinement algorithm

In the previous section, we considered the case where the refinement algorithm has ended up with a Lyapunov function. However, depending on the starting grid of collocation

points, the algorithm might terminate without constructing a Lyapunov function. More precisely, the constructed function v from the last refinement step could still have some patches, where the orbital derivative is positive. The reason for these patches is that the last refinement step placed all Voronoi vertices in areas where $v'(x) < 0$, thus missing the areas where refinement would be necessary. In Chapter 4, an improved refinement algorithm is introduced to add points to the grid only in these areas.

The following section illustrates the effect of choosing different starting grids on the output of the refinement algorithm. For each example, a table is provided to list the following information:

- The initial grid to start the refinement process $X_{initial}$.
- The final grid of points obtained after the refinement terminated X_{final}.
- The value of the fill distance h for the initial grid. We have considered different values of h until we have reached the case where the refinement did not add any more points, and at the same time no small patches are remaining.
- The time needed in each case to construct an RBF approximant after the whole refinement process (time 1)
- The time for calculating and plotting the orbital derivative of the constructed RBF approximant with X_{final} (time 2).
- Finally, we have highlighted the case where we have successfully construct a Lyapunov function with fewest collocation points and least time.

Example 3.5. *Considering again Example 3.2, we have started with regular grids of* 16 *to* 360 *points in* $K = [-1, 1]^2$. *In Table 3.1 we have listed the value of h for the initial grid and the corresponding number of points in the initial grid* $N_{initial}$ *as well as the number of points in the grid after the refinement algorithm has terminated. Moreover, it shows the running time for solving the linear systems in all refinement steps (time 1) and for calculating and plotting the orbital derivative of the constructed Lyapunov function for the final set of grid points (time 2), note that this time is proportional to the number* N_{final}

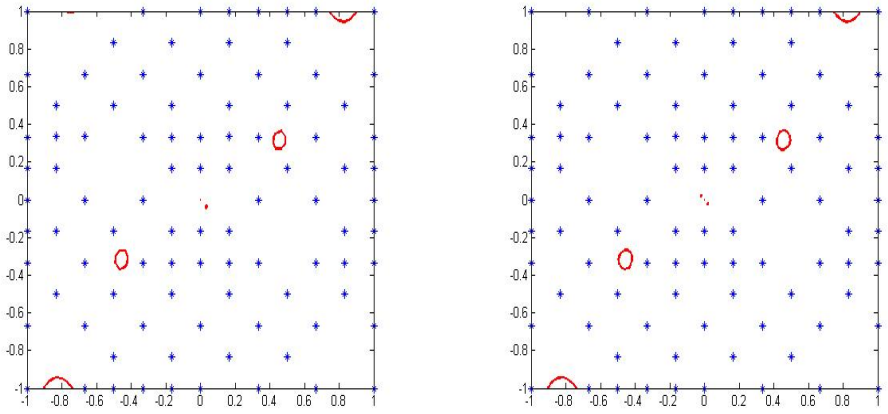
of grid points in the final grid. For all calculations we have used a standard laptop with an Intel(R) Core (TM) i5-3550 CPU @ 3.30 GHz processor.

h	$N_{initial}$	N_{final}	time 1 (refinement)	time 2 (plot)
2/3	16	93 (patches)	4.5 sec.	14.8 sec.
1/2	24	88	2.1 sec.	13.9 sec.
2/5	36	104 (patches)	10 sec.	16.4 sec.
1/3	48	94 (patches)	4.4 sec.	14.8 sec.
1/4	80	86 (patches)	1.5 sec.	13.6 sec.
1/5	120	124 (patches)	3.4 sec.	19.5 sec.
1/6	168	168 (patches)	1.2 sec.	26.5 sec.
1/7	224	224 (patches)	2.2 sec.	35 sec.
1/8	288	288 (patches)	4 sec.	45 sec.
1/9	360	360	6 sec.	1 min.

Table 3.1: The number of grid points and the value of h, we have used to construct Lyapunov functions for Example 3.2. This table shows the initial sets of regular grid points and the final number of points after refinement. Moreover, it shows the running time for solving the linear systems in all refinement steps (time 1) and for calculating and plotting the orbital derivative of the constructed Lyapunov function for the final set of grid points (time 2). "Patches" means that after termination of the refinement algorithm there are still areas with positive orbital derivative remaining. **The shortest successful construction of a Lyapunov function is achieved with an initial grid of 24 points**.

For small (16) and larger (36, 48, 80, 120, 168) numbers in our starting grid, the refinement algorithm stopped, but there were patches of areas where $v'(x, y) > 0$, see, for example, Figure 3.13. On the other hand, when we started with 24 grid points, we achieved good results with no patches remaining at the end, see, for example, Figure 3.14. There were also some cases, for 224 and 288 starting points, where the refinement algorithm did not add any points but we still had small patches. Finally, with 360 points, we reached the case where the refinement did not add any points and at the same time we did not have small patches of positive orbital derivatives in $K \setminus E_{nh}$, where $E_{nh} = [-0.1, 0.1]^2$.

For $X_{initial}$ =24 and 360 we have checked the sign of v' on the grid $X_{check} = \{(x, y) \in$



(a) The refinement with 16 initial grid points.

(b) The refinement with 48 initial grid points.

Figure 3.13: (a) The level set $v'(x, y) = 0$ of the orbital derivative after the last refinement step started with 16 points and ended up with 93 points, (b) the level set $v'(x, y) = 0$ of the last refinement step started with 48 points and ended up with 94 points. In both cases we can see the small patches remaining at the end of the refinement procedure, where the orbital derivative is positive (red areas).

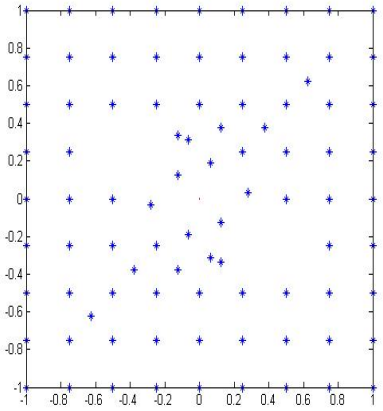


Figure 3.14: The final grid points after the last refinement step started with 24 points and ended up with 88 points. As we can see, there are no small patches remaining at the end of the refinement procedure.

$\mathbb{R}^2 \mid x, y \in \{0, \pm h_{check}, \ldots, \pm 1\}\} \setminus E_{nh}$ *with* $h_{check} = 10^{-3}$.

The refinement algorithm solves (small) linear systems in each refinement step, while for the grid with 360 points only one (larger) linear system needs to be solved (time 1). Comparing the successful construction of a Lyapunov function (24 and 360 initial points), the starting grids with 24 has a shorter time 1 than the grid with 360 points, where no refinement step is necessary, so here solving several small systems is faster than solving a large one. Moreover, as the number of points in the final grid is proportional to the time to calculate and plot the orbital derivative (time 2), which takes much longer than solving the linear systems, smaller starting grid (24) takes a shorter overall time (time 1+ time 2).

h	$N_{initial}$	N_{final}	time 1 (refinement)	time 2 (plot)
0.6	16	64 (patches)	1.2 sec.	9.6 sec.
0.45	24	87 (patches)	3.3 sec.	13 sec.
0.36	36	88	1.8 sec.	13 sec.
0.3	48	60 (patches)	0.4 sec.	9 sec.
0.225	80	104	1.7 sec.	15.5 sec.
0.18	120	132	4.5 sec.	19.5 sec.
0.15	168	168	1.2 sec.	25 sec.

Table 3.2: The number of grid points and the values of h we have used to construct Lyapunov functions for Example 2.10 of [17]. This table shows the initial sets of regular grid points and the final number of points after refinement. Moreover, it shows the running time for solving the linear systems in all refinement steps (time 1) and for calculating and plotting the orbital derivative of the constructed Lyapunov function for the final set of grid points (time 2). The shortest successful construction of a Lyapunov function is achieved with an initial grid of 168 points, without any refinement step (time 1); **the shortest construction and plot (time 1+ time 2), however, is achieved with an initial grid of 36 points**.

Example 3.6. *A similar behaviour was observed in Example 3.3, where we have started with regular grids of* 16 *to* 168 *points, see Table 3.2. We used* $X_{check} = \{(x, y) \in \mathbb{R}^2 \mid$

$x, y \in \{0, \pm h_{check}, \ldots, \pm 0.9\}\} \setminus E_{nh}$ *where* $E_{nh} = [-0.1, 0.1]^2$ *and* $h_{check} = 10^{-3}$ *to check that the sign of* v' *is negative for the examples with* 36, 80, 120 *and* 168 *starting points. In this example, we reached the case where the refinement algorithm did not add more points and we did not have patches remaining, with 168 regular grid points, see Table 3.2.*

Although in this example, the shortest time to solve the linear system(s) is achieved with the grid of 168 points, without any refinement, again the sum of time 1 and time 2 (solving linear system(s) and plotting the orbital derivative) is the longest for the initial grid of 168 points.

Example 3.7. *The influence of the starting grid on the refinement algorithm was also investigated in the 3-dimensional system introduced in Example 3.4. In Table 3.3, we have presented the values of h and the corresponding number of regular starting grid points, the final number of grid points after the refinement process and the running time in each case.*

h	$N_{initial}$	N_{final}	time 1 (refinement)	time 2 (plot)
0.45	124	180 (patches)	5 sec.	25 min.
0.3	342	458	50 sec.	1 hr. 15 min.
0.225	728	728 (patches)	23.5 sec.	2 hrs.
0.18	1330	1330 (patches)	1 min. 20 sec.	3 hrs. 30 min.
0.15	2196	2196	4 min.	6 hrs. 47 min.

Table 3.3: The number of grid points and the values of h we have used to construct Lyapunov functions for Example 3.4. The table shows the number of regular grid points we started with and the final number of points after refinement. Moreover, it shows the running time for solving the linear systems in all refinement steps (time 1) and for calculating and plotting the orbital derivative of the constructed Lyapunov function for the final set of grid points (time 2). **The shortest successful construction of a Lyapunov function is achieved with an initial grid of 342 points**.

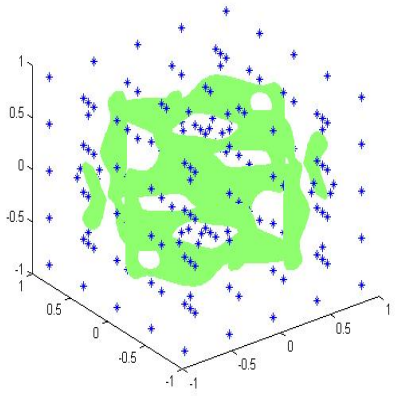
Again, when we started with a coarse grid (124 initial points), the refinement algorithm stopped but with some remaining patches where $v'(x, y, z) > 0$*, see Figure 3.15 (a). Moreover, starting with a finer grid of 728 or 1330 initial points, the refinement algorithm did not add any more points but we still have patches, see Figure 3.15 (b) and (c). However,*

starting with a regular grid of 342 points, and after 4 refinement steps, we ended up with a desirable result with no patches remaining at the end. A similar result was achieved with a regular fine grid of 2196 points, except for a small neighborhood of the origin, see Figure 3.16. Recall that, for this example, we used $X_{check} = \{(x, y, z) \in \mathbb{R}^3 \,|\, x, y, z \in \{0, \pm h_{check}, \ldots, \pm 0.9\}\} \backslash E_{nh}$ *where* $E_{nh} = [-0.2, 0.2]^3$ *and* $h_{check} = 10^{-2}$ *to check that the sign of* v' *is negative everywhere for 342 and 2196 points in the initial grid.*

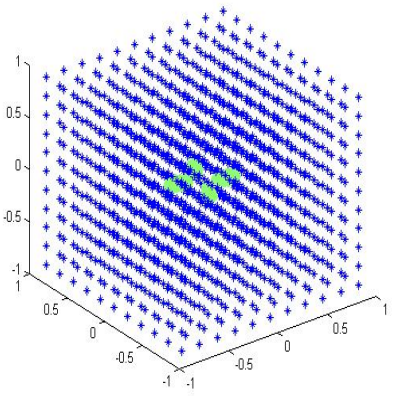
Here, there is a considerable difference between the two successful constructions of more than a factor four, both in the times to solve the linear systems (time 1) and in the time to plot the orbital derivative (time 2).

Summarising, starting with a grid which is too coarse or too fine may end the refinement algorithm with a function that has still areas where the orbital derivative is positive. A moderately coarse grid is the most promising starting point. In most cases, the time to calculate a Lyapunov function was shorter by using the refinement algorithm, even if it involves solving a system of linear equations in each refinement step (time 1). An even greater advantage of the refinement algorithm, however, becomes apparent when using the calculated Lyapunov function further, e.g., to calculate and plot its orbital derivative (time 2), which is proportional to the number of points in the final grid; here, the reduction in the number of grid points pays off considerably.

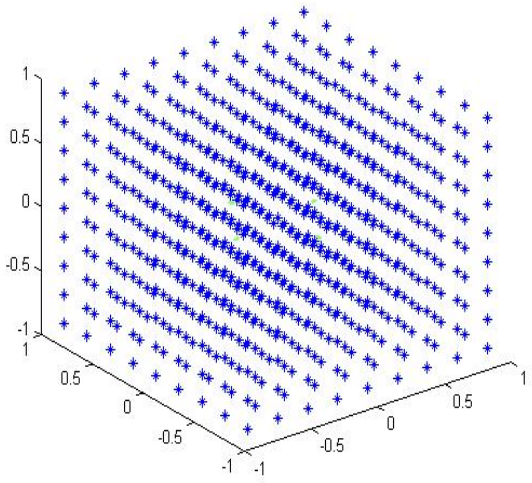
In general, the complexity to solve a linear system (2.7) is of order $\mathcal{O}(N^3)$ *and it is of order* $\mathcal{O}(MN)$ *for computing and verifying the negativity of the orbital derivative of an RBF function, where* N *is the number of grid points used for the calculation of the RBF function and* M *is the number of grid points used for the evaluation and the verification process. However, if we parallelize the evaluation \ verification step to be executed on* M *parallel processors independently, then this step will require much shorter time than solving the linear system.*



(a) The refinement with 124 initial grid points.



(b) The refinement with 1330 initial grid points.



(c) The refinement with 728 initial grid points (4 little green patches).

Figure 3.15: (a) The level set $v'(x, y, z) = 0$ of the orbital derivative after the last refinement step started with 124 points and ended up with 180 points, (b) and (c) the level set $v'(x, y, z) = 0$ of the orbital derivative calculated with 728 and 1330 points, where the refinement did not add more points. In all cases we can see some patches remaining at the end (green), where $v'(x, y, z) > 0$.

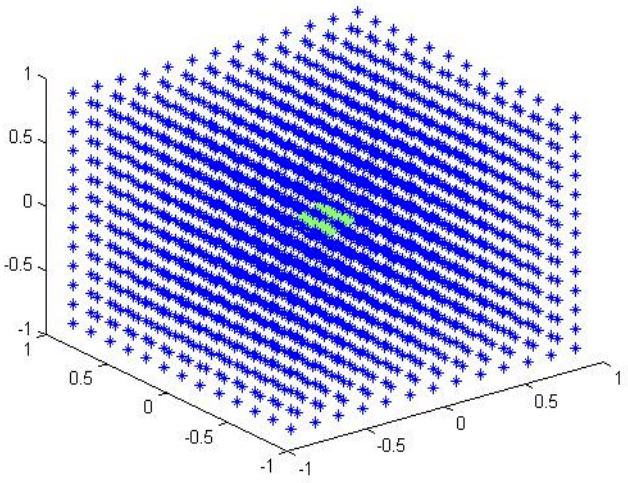


Figure 3.16: The level set $v'(x, y, z) = 0$ of the orbital derivative calculated with a regular grid of 2196 points. As we can see there are no patches except for a small neighbourhood, $[-0.2, 0.2]^3$, of the equilibrium point (0,0,0).

Chapter 4

The Modified Grid Refinement Algorithms

During the investigation of the influence of the starting grid on the refinement algorithm, we found out that some of the starting grids may produce a function which still has positive orbital derivative in some areas. Mainly, we do not know in advance which number of the starting grid points ends the refinement process with a function that has negative orbital derivative everywhere. Therefore, we need, somehow, to find a way to keep adding points to the grid after the early termination of the refinement algorithm.
In our case, we have a cloud of points ($x \in K$) with positive orbital derivative, clustered into patches based on their distances. Since our target is to add a point in every patch, this makes the centres of these clusters perfect candidates to be added to our grid of collocation points. Therefore, to deal with this situation, we have designed two extension algorithms that allow to add points in the patches where the constructed function has positive orbital derivative. The main tool of these algorithms is using the data clustering methods that produce cluster centres. We call these algorithms *"extension algorithms"* because they extend the work of the refinement which is adding points to our grid only where we need to do so. Thus, the modified grid refinement algorithm is the refinement algorithm of Section 3.1, combined with an extension algorithm.

The first section of this Chapter gives a brief introduction to data clustering analysis.

Then, the strategy of the first and second extension algorithms along with numerical examples, are presented in the second and third sections, respectively.

4.1 Introduction to data clustering

Data clustering is the process of classifying a set of data points into clusters which are sharing a similarity criterion. The similarity criterion varies according to the data types to be clustered. For example, if the data are points in a d-dimensional space then the similarity criterion will use a distance function to group them. However, biological data will be clustered according to similar sequences or networks, whereas multimedia data such as images or videos may define similarity based on similar music or photographs [1]. As a result, numerous algorithms have been designed to serve the different objectives of clusterings. These clustering algorithms can be distinguished based on their

1. Technique:
    - **Hierarchical clustering**, builds a hierarchy of clusters (i.e., nested clusters).
    - **Partitioning clustering**, builds a unique partitioning of the data set.
2. Approach:
    - **Fuzzy (soft) clustering**, objects can belong to one or more clusters at the same time as each object is assigned to clusters according to its level of membership.
    - **Crisp (hard) clustering**, each object is assigned to a unique cluster.
3. Mode:
    - **On-line clustering**, the algorithm gets the objects in the data set in order, so the cluster centres are adjusted every time a new object is introduced.
    - **Off-line clustering**, the algorithm gets the whole data set at once, and thus the cluster centres are computed at once.

This section aims to partition the points $x \in K$ where $v'(x) > 0$ into unique patches (clusters), then find the centre of each patch. Thus, we are concerned with clustering algorithms that are used in partitioning technique, applied in off-line mode as crisp approach.

The most representative off-line clustering techniques are k-means, fuzzy C-means, mountain and subtractive clustering [40, 46, 28]. However, we only make use of two algorithms namely k-means clustering and subtractive clustering. We dismiss the other algorithms as the fuzzy C-means applies fuzzy approach (not crisp) and the mountain method is just the original form of the subtractive clustering.

4.1.1 The k-means clustering

Let $(x_1, x_2, \ldots, x_m)$ be m objects in a data set, where each object is a d-dimensional vector, i.e., $x_j \in \mathbb{R}^d$, for all $j = 1, 2, \ldots, m$. Then, the k-means algorithm groups the m objects into k clusters, where k is a positive number fixed a *priori*. The k-means algorithm assigns each object $x_j,\ j = 1, \ldots, m$ to the cluster with nearest centroid.

The steps of the k-means algorithm:

1. Initialize the cluster centres $c_i,\ i = 1, \ldots, k$ which is done by the algorithm.
2. Calculate the distance between each object $x_j,\ j = 1, \ldots, m$, in the data set to the cluster centres.
3. Group the objects to clusters based on the minimum distance to all centres

$$\mu_i = \{x_j \mid \|x_j - c_i\| \leq \|x_j - c_l\|,\ l \neq i,\ j = 1, \ldots, m\}.$$

4. For each cluster $i = 1, \ldots, k$, recalculate the new cluster center using the mean of all data points in the cluster

$$c_i = \frac{1}{|\mu_i|} \sum_{j \in \mu_i} x_j,\ \ \forall i.$$

   where $|\mu_i|$ = number of elements in the set μ_i.

5. Iterate until no object was reassigned.

4.1.2 The Subtractive clustering

The subtractive clustering method is a modified version of the mountain clustering method proposed by Yager and Filev in 1993 [63]. The mountain clustering method is a simple

and effective method to estimate cluster centres by first forming a grid on the data space and then constructing a density measure called the mountain function on each grid point (i.e. in the intersection of the grid lines) in which they are considered as potential cluster centres. However, it becomes computationally expensive in higher dimensional spaces due to the evaluation of the mountain function over all grid points [59].
Therefore, an alternative approach, the subtractive clustering, has been developed by Chiu in 1994 [11] to solve this problem. The main concept of the subtractive clustering method is that it considers the data points themselves (objects) as candidates to be cluster centres but not the grid points as in the mountain method. This means that the computation of the subtractive method is not proportional to the dimension of the problem any longer, but to the size of it [46, 28].

The steps of the subtractive algorithm:

1. Assume that each point in the data set is a candidate to be a cluster centre. Then, for each point $x_i,\ i = 1, \ldots, m$, calculate a density function of the form
$$D_i = \sum_{j=1}^{m} \exp\left(- \frac{\|x_i - x_j\|^2}{(\frac{R}{2})^2} \right) \tag{4.1}$$
where $R > 0$ is a constant representing the radius of the neighbourhood area.

2. Based on the density of the neighbouring data points, within the area determined by R, the algorithm selects the point x_i with the largest density value D_i to be the first cluster centre.

3. All the data points in the neighbourhood of radius R of the first cluster centre will be eliminated in order to find the next potential cluster centre.

4. The algorithm will iterate this process until a sufficient number of cluster centres is obtained, and all the points are assigned to the cluster with nearest centre. The values of the radius R usually lie within $[0.2, 0.5]$ as reported in the literature [53]. Note that, if the radius is too small, then too many cluster centres will be generated and if it is too big, then the algorithm will generate few cluster centres.

In [28], a comparison between the performance of the four off-line clustering algorithms, k-means clustering, fuzzy C-means clustering, mountain clustering and subtractive clus-

tering, was presented. They were tested on a medical problem related to heart disease diagnosis. In Table 4.1, we state the main differences between the k-means and the subtractive clustering methods, in terms of determining the cluster centres.

K-means clustering	Subtractive clustering
The centres returned are different based on the initial positions of the cluster centres which are chosen by the algorithm (not unique).	The centres returned are fixed (unique).
The number of clusters has to be specified a priori	The number of clusters is done automatically based on the value of the radius specified.

Table 4.1: The difference between the k-means and the subtractive clustering methods.

4.2 The modified grid refinement algorithms

The modified grid refinement algorithms consist of two iterative steps performed in order until a successful construction of a Lyapunov function is achieved. The steps are:

A The whole refinement process until it terminates.

B An extension process which links two refinement procedures by adding points to the grid when the refinement fails to do so.
The extension process can be done through two algorithms: the first one uses the Delaunay triangulation along with the k-means clustering to add grid points, whereas the second algorithm uses the subtractive clustering. These extension algorithms will be explained in details within the steps of the modified algorithms.

4.2.1 The 1st modified algorithm: using Delaunay and k-means

Remember that our goal is to find the centres of the patches where the orbital derivative is positive. For the extension step of the first modified grid refinement algorithm we will

apply the first extension technique which employs the Delaunay triangulation and the k-means clustering to place grid points. The reason for using the Delaunay triangulation is that the clustering returned by the k-means algorithm depends on the positions of the initial cluster centres. Since those centres are selected automatically by the algorithm, we would get different results every time we run the algorithm. Therefore, instead of running the algorithm several times to find the k optimal centres for our problem, we run the algorithm once to find the centre for each cluster (patch) individually. This will be done by partitioning the set K into simplices by the Delaunay triangulation, and then specifying each cluster according to the simplex it belongs to.

The Algorithm Strategy:

- A1: A refinement process

  Let v_n be the constructed function from the unsuccessful step of the refinement algorithm, such that $v_n'(x) > 0$ at some points $x \in K \subset \mathbb{R}^d$. Moreover, let X_n be the final grid points obtained before the termination of the refinement process, then

- B1: An extension process

  1. Generate a Delaunay triangulation (see Section 3.1.1) for X_n, where the vertices of the simplices are the grid points in X_n.
  2. Fix a test grid $X_{test} \subset K \subset \mathbb{R}^d$ with density specified by h_{test}, to check the sign of the orbital derivative of v_n at each point of X_{test}.
  3. Build an array with the points $x \in X_{test}$, which have positive orbital derivative, i.e., $PO = \{x \in X_{test} \mid v_n'(x) > 0\}$, excluding a specified neighbourhood of the equilibrium point.
  4. Cluster the points in PO according to the simplex they belong to, then use the *k-means* MATLAB function to calculate the centre of each cluster.
  5. Add all the cluster centres into array CR.
  6. Run a *closeness test* on every pair of centres in CR for two reasons:
     - To avoid adding too close centres, which may cause the collocation matrix to be nearly singular.

– To avoid adding more than one centre for a single cluster in the cases where a patch is spread into 2 or more different simplices, see Example 4.1.

The closeness test:

(a) Fix a parameter d =the minimal Euclidean distance between pairs of grid points in X_n.

(b) Calculate d_c =the minimal Euclidean distance between pairs of centres in CR.

(c) If $d_c \leq \frac{1}{2}\, d$, then find the pairs of centres that produced this d_c.

(d) Remove one centre of each pair.

(e) Continue until $d_c \geq \frac{1}{2}\, d$.

7. Add the remaining centres to the existing grid points X_n.

8. Calculate the function v_{n+1} using the RBF method with $X_{n+1} = X_n \cup CR$.
   If
   $v'_{n+1} < 0$ on K $\Rightarrow$ STOP (by checking the sign of v'_{n+1} in X_{test}).
   else

- A2: A refinement step

  If v_{n+1} still has areas where $v'_{n+1} \geq 0$, then we perform steps 3-6 of the refinement algorithm, introduced in Section 3.1.2, to check if it adds more points and continue until it terminates again. If v_{final}, the function constructed with the final set of grid points X_{final} obtained after the termination of the current refinement step, still has positive orbital derivative at some points in K, then

- B2: An extension process

  We apply the extension algorithm again (steps 1-8).

Iterate this process (repeat steps A and B constantly) until we successfully construct a Lyapunov function on K. In the next section, we will illustrate the performance of the first modified grid refinement algorithm on numerical examples.

4.2.2 Numerical Examples

Now, we will apply our first modified refinement algorithm to the same examples investigated in Section 3.3 and show how the algorithm solves the problem of the early termination of the refinement algorithm without constructing a Lyapunov function. Note that, for all examples in this Section and Section 4.2.4, we exclude a small neighbourhood of the equilibrium from adding grid points. Moreover, after successfully constructing a Lyapunov function with the modified algorithm, we run a preliminary checking of the negativity of the orbital derivative on a grid X_{check}. Later, we will carry out a further and more accurate checking in Chapter 6. Finally, a summary table is presented after each example to point out the number of grid points added after each step as well as the computation time of the whole process.

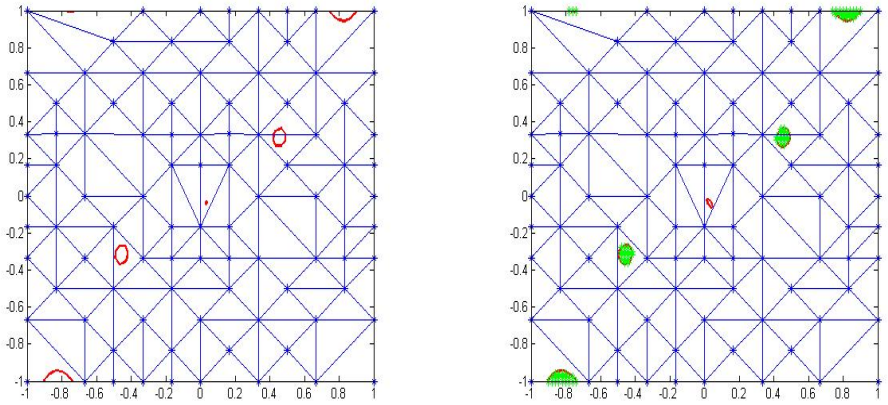
Example 4.1. *Again, we consider the system in Example 3.2. According to Table 3.1, the refinement algorithm terminates, with patches remaining at the end, with the starting points* $N_{initial} = 16, 36, 48, 80, 120, 168, 224$ *and* 288. *In this example, we will examine the algorithm on the case where we started with* $N_{initial} = 16$ *regular grid points.*

1. *A1: Refinement process*

   In this case, the refinement algorithm terminates after 9 refinement steps with $N_9 = 93$ *points. Figure 3.13 (a), shows the level set of* $v_9'(x, y) = 0$, *with the (red) areas where* $v_9' > 0$.

2. *B1: 1st extension process*

   - *The first step of the algorithm is to generate a Delaunay triangulation for the final set of grid points* $X_9 = \{x_i\}_{i=1}^{93}$, *as shown in figure 4.1 (a).*
   - *Then, we check the sign of the orbital derivative of the constructed funtion* $v_9(x, y)$ *over a test grid* $X_{test} = \{x, y \in \mathbb{R}^2 \mid x, y \in \{0, \pm h_{test}, \ldots, \pm 1\}\} \backslash [-0.1, 0.1]^2$ *with* $h_{test} = 0.02$, *which produces the array* $PO_1 = \{(x, y) \in X_{test} \subset \mathbb{R}^2 \mid v_9'(x, y) > 0\}$, *containing 74 points, green (*) in figure 4.1 (b).*
   - *Clustering the 74 points in* PO_1 *into small arrays according to which triangle of the Delaunay triangulation they lie in, gives 6 arrays* CL_i, $i = 1, \ldots, 6$.



(a) The Delaunay triangulation of X_9.

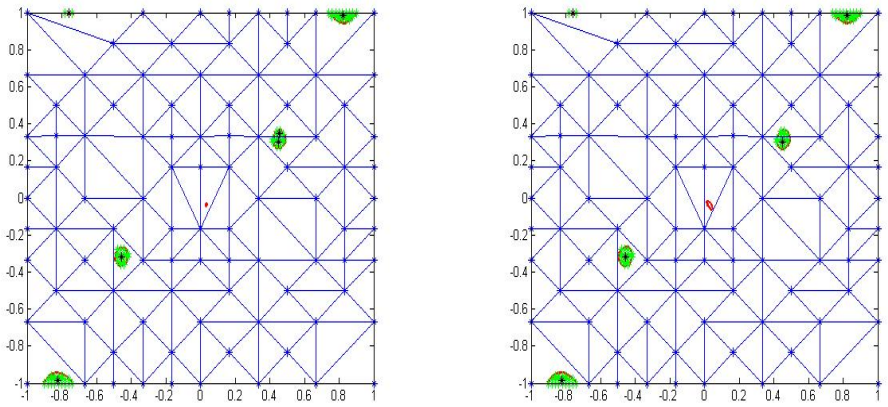
(b) Clustering the points in PO_1 into 6 clusters.

Figure 4.1: (a) The Delaunay triangulation for X_9, where the vertices of the triangles are our 93 grid points (blue *), (b) the points in PO_1 (green *) located in the areas where the orbital derivative is positive and grouped into 6 clusters based on the triangles they belong to.

- *For each cluster array CL_i, $i = 1, \ldots, 6$, we calculate its centre using the k-means MATLAB function, see figure 4.2 (a), then we add all the 6 centres into array CR_1.*
- *We run the closeness test on CR_1:*
  $d = 0.1667$, $d_c = 0.0444 < \frac{1}{2}d = 0.08335$, and the pair of centres that produced d_c are the 5th and 6th centres. Thus, we remove the 6th centre from array $CR_1 \Rightarrow CR_1 = 5$ centres, as shown in figure 4.2 (b).
- *Adding the 5 centres to the existing grid points X_9 gives a new set of grid points $X_{10} = X_9 \cup CR_1 = \{x_i\}_{i=1}^{98}$.*
- *Calculate the function v_{10} for the new set of grid points X_{10}. As we see in figure 4.3, the function v_{10} still has positive orbital derivative in some areas, meaning that we need to add more points to the grid.*

3. *A2: Refinement process*

- *The refinement algorithm with X_{10} did not add any more points, thus we need to perform another extension step.*



(a) The 6 clusters centres. (b) The clusters centres to be added.

Figure 4.2: (a) The clusters centres (black *) calculated using the k-means MATLAB function, (b) the remaining clusters centres after running the closeness test and removing the 6th centre.

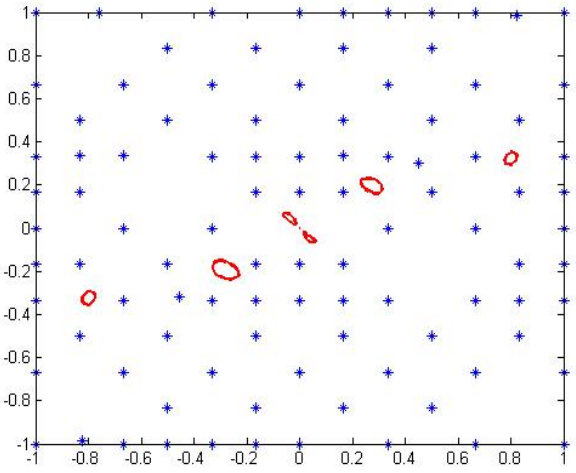


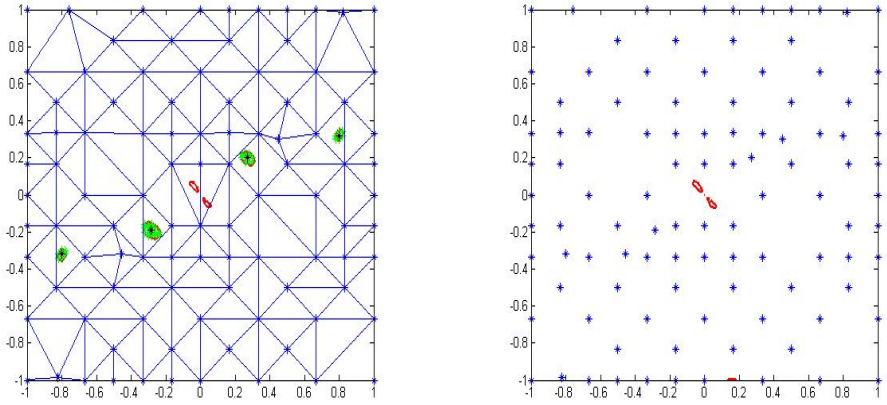
Figure 4.3: The level set $v'_{10}(x, y) = 0$ of the orbital derivative calculated with the new set of grid points X_{10}. As we can see there are still patches remaining where the orbital derivative is positive.

4. B2: 2nd extension process

- *After going through the same steps again, we obtain a new clusters centres $CR_2 = 4$ centres as shown in figure 4.4 (a), and the closeness test yields:*

$d = 0.1203$, $d_c = 0.5318 > \frac{1}{2}d = 0.06015$, *therefore no centres need to be removed.*

- *Adding the 4 centres to the previous grid point* X_{10} *gives a new set of grid points* $X_{11} = X_{10} \cup CR_2 = \{x_i\}_{i=1}^{102}$.
- *Calculate the function* v_{11} *for* X_{11}*, which also still has some areas where the orbital derivative is positive, see figure 4.4 (b).*



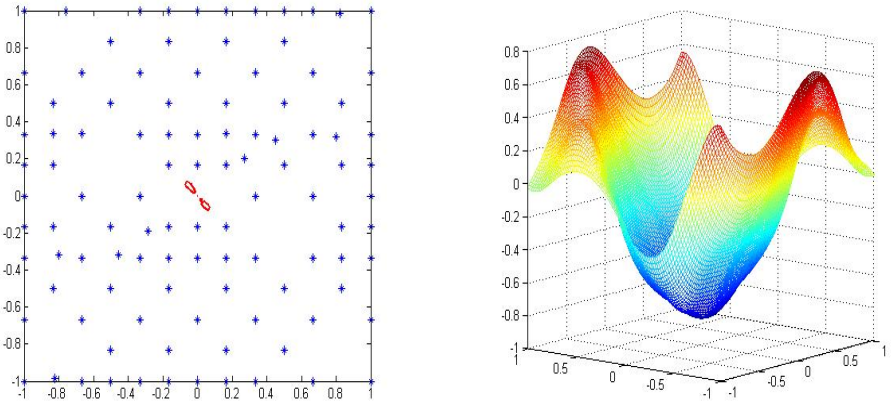
(a) The 4 clusters centres to be added. (b) The level set of $v'_{11}(x, y) = 0$.

Figure 4.4: (a) The clusters centres (black *) calculated in the second extension step, (b) the level set of $v'_{11}(x, y) = 0$ of the constructed function v_{11} with the grid X_{11} of 102 points. The patches where the orbital derivative is positive (red areas).

5. A3: Refinement process

- *The refinement algorithm with* X_{11} *adds 2 more points to the grid giving a new set* $X_{12} = X_{11} \cup \{y_j\}_{j=1}^{2} = \{x_i\}_{i=1}^{104}$.
- *Finally, we were able to construct a Lyapunov function* v_{12} *with* $N_{final} = 104$ *grid points, as shown in figure 4.5 (a). The function* v_{12} *has negative orbital derivative on a checking grid* X_{check} *with* $h_{check} = 10^{-2}$.

Table 4.2 summarizes the performance of the first modified grid refinement applied to Example 3.5. The refinement algorithm started with 16 regular grid points and stopped with 93 points, without constructing a Lyapunov function. However, with the modified



(a) The grid points $X_{12} = 104$.

(b) The constructed Lyapunov function $v_{12}(x, y)$.

Figure 4.5: (a) The level set $v'_{12}(x, y) = 0$ of the constructed function v_{12} with the final set of grid points X_{12}, no areas of positive orbital derivative remaining except for in a small neighbourhood $E_{nh} = [-0.1, 0.1]^2$, of the equilibrium point $(0, 0)$, (b) the constructed Lyapunov function $v_{12}(x, y)$ with the first extension algorithm.

algorithm, we have managed to keep refining the grid until successfully constructing a Lyapunov function with 104 grid points.

	Steps preformed in order	No. of points added	N	Time (whole process)
A1	refinement	77	93	45.66 sec.
B1	extension	5	98	3 min. 43 sec.
A2	refinement	0	98	1 sec.
B2	extension	4	102	3 min. 45 sec.
A3	refinement	2	104	47.12 sec.
				total= 9 min.

Table 4.2: The steps of the first modified grid refinement algorithm performed in sequence, with the number of points added after each step as well as the total number of grid points N. The table also shows the whole time needed for each process, including the steps of the algorithm + solving the linear system for the obtained set of grid points + plotting the orbital derivative of the corresponding RBF approximants.

Example 4.2 (Example 2.10)**.** *Consider another 2-dimensional example, namely the system in Example 3.3. For this system, the refinement algorithm terminates, without constructing a Lyapunov function, when it started with $N_{initial} = 16, 24, 48$ regular grid points, see Table 3.2.*

In this example, we examine the first modified algorithm in the case where the refinement started with $N_{initial} = 48$ regular grid points.

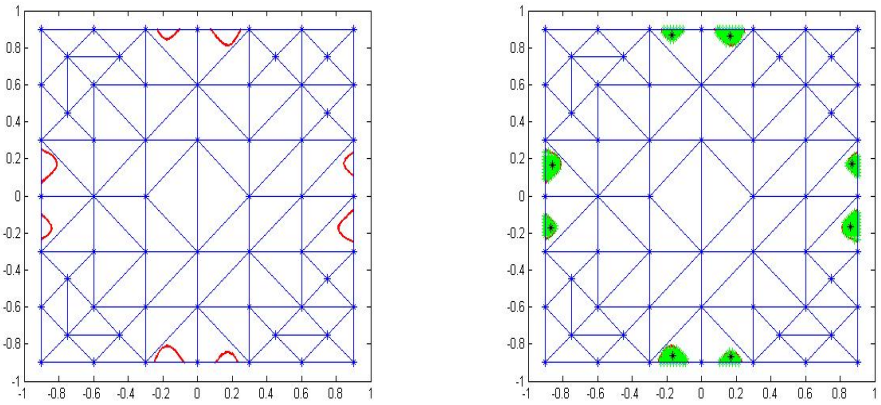
1. *A1: Refinement process*

   The refinement algorithm terminates after only one step with $N_1 = 60$ points. Figure 4.6 (a), shows the level sets of $v_1'(x, y) = 0$, where the (red) areas are the patches where $v_1'(x, y) > 0$.

2. *B1: 1st extension process*

   - *Generate a Delaunay triangulation for the final set of grid points obtained from the previous refinement step $X_1 = \{x_i\}_{i=1}^{60}$, see figure 4.6 (a).*
   - *Fix a test grid X_{test} with $h_{test} = 0.013$, to check the sign of $v_1'(x, y)$, then build the array $PO_1 = \{(x, y) \in X_{test} \subset \mathbb{R}^2 \mid v_1'(x, y) > 0\}$, the green (*) in figure 4.6 (b).*
   - *The points in PO_1, will be clustered into 8 clusters according to the triangle they belong to. Therefore, we are going to group the points in each cluster (triangle) into 8 different arrays $CL_i, i = 1, \ldots, 8$.*
   - *We calculate the centre of each cluster, represented by an array CL, using the k-means MATLAB function, black (*) in figure 4.6 (b).*
   - *Add all the 8 centres into array CR_1 for the closeness test $d = 0.2121$, $d_c = 0.3366 > \frac{1}{2}d = 0.106$, therefore no centres need to be removed.*
   - *Adding the 8 centres into the existing set of grid points X_1 gives a new set of grid points $X_2 = X_1 \cup CR_1 = \{x_i\}_{i=1}^{68}$.*
   - *Construct an RBF function v_2 for the new set of grid points X_2. The function v_2 still has areas where $v_2'(x, y) > 0$ as shown in figure 4.7. Thus we need to add more points to the grid.*

3. *A2: Refinement process*



(a) The Delaunay triangulation of X_1. (b) The 8 clusters and their centres.

Figure 4.6: (a) The Delaunay triangulation for X_1, where the vertices of the triangles are our 93 grid points (blue *), (b) the points in PO_1 (green *) located in the areas where the orbital derivative is positive and grouped into 8 clusters based on the triangles they belong to. The black (*) represent the centres of the clusters calculated by the k-means function.

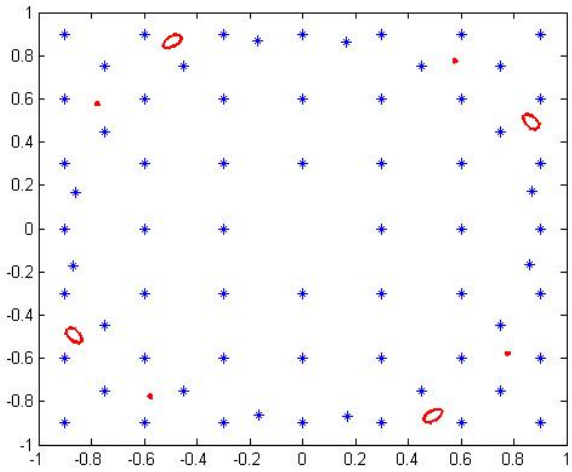
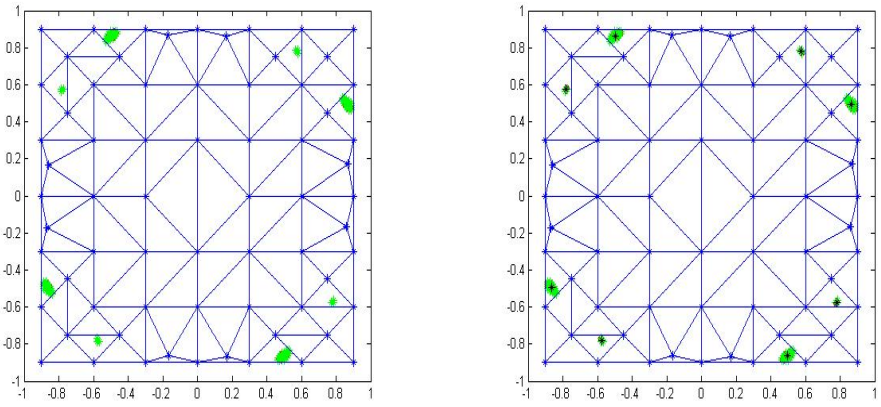


Figure 4.7: The level set $v_2'(x, y) = 0$ of the orbital derivative calculated with the new set of grid points X_2. As we can see there are still patches remaining where the orbital derivative is positive (red).

- *The refinement algorithm with X_2 terminates without adding any more points to the grid. Therefore, another extension process needs to be performed.*

4. *B2: 2nd extension process*

- *Generate a Delaunay triangulation for X_2, as shown in figure 4.8 (a) .*
- *Fix a test grid X_{test} with $h_{test} = 0.013$, to check the sign of $v_2'(x, y)$, then obtain the array $PO_2 = \{(x, y) \in X_{test} \subset \mathbb{R}^2 \mid v_2'(x, y) > 0\}$.*
- *We go through the same steps and obtain a new set of cluster centres $CR_2 = 8$ centres, black (*) in figure 4.8 (b). The closeness test of CR_2 yields $d = 0.1324$, $d_c = 0.4063 > \frac{1}{2}d = 0.066$, therefore no centres need to be removed.*



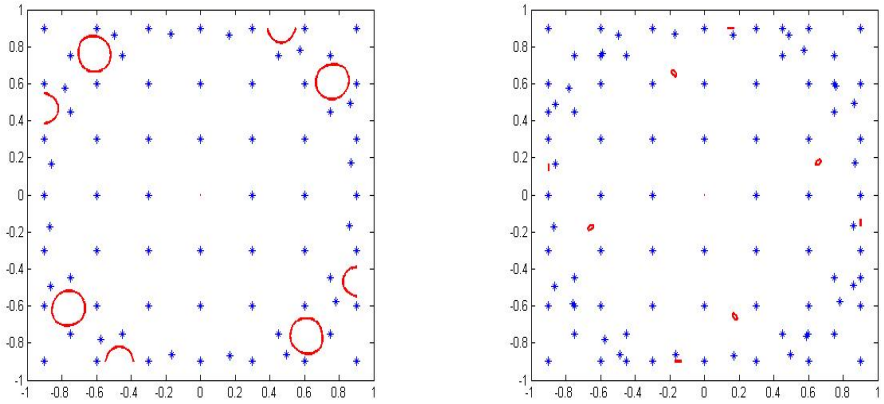
(a) The Delaunay triangulation of X_2. (b) The 8 clusters and their centres.

Figure 4.8: (a) The Delaunay triangulation for X_2, where the vertices of the triangles are our 93 grid points (blue *), (b) the points in PO_2 (green *) located in the areas where the orbital derivative is positive and grouped into 8 clusters based on the triangles they belong to. The black (*) represent the centres of the clusters calculated by the k-means function.

- *Add the new centres to the previously obtained set of grid points X_2 to obtain a new set $X_3 = X_2 \cup CR_2 = \{x_i\}_{i=1}^{76}$.*
- *The constructed RBF function v_3 with X_3 still has positive orbital derivative in some areas, see figure 4.9 (a).*

5. *A3: Refinement process*

- *Performing one step of the refinement algorithm with X_3 adds 16 grid points, giving a new set $X_4 = X_3 \cup \{y_j\}_{j=1}^{16} = \{x_i\}_{i=1}^{92}$. However, the constructed function v_4 with X_4 still has positive orbital derivative in some areas, see figure 4.9 (b).*



(a) The level sets of $v_3'(x, y) = 0$.

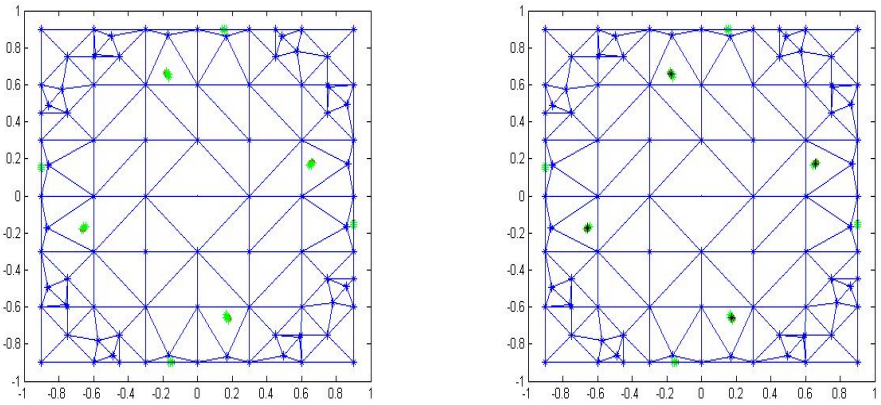
(b) The level sets of $v_4'(x, y) = 0$

Figure 4.9: (a) The level sets of the orbital derivative of the constructed function v_3 with X_3. The (red) patches are the areas where $v_3'(x, y) > 0$, (b) the level sets of $v_4'(x, y) = 0$ of the constructed function with X_4, with the red patches where $v_4'(x, y) > 0$.

6. *B3: 3rd extension process*

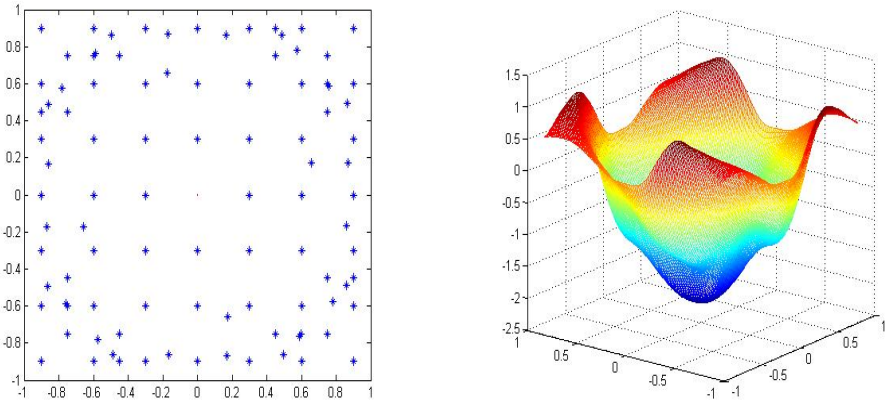
- *Repeating the same steps gives a new set of cluster centres $CR_3 = 4$ centres and the closeness test on CR_3 yields*
  $d = 0.0154$, $d_c = 0.9617 > \frac{1}{2}d = 0.0077$, *therefore no centres need to be removed.*
- *Add the 4 centres to X_4 to obtain a new set of grid points $X_5 = X_4 \cup CR_3 = \{x_i\}_{i=1}^{96}$.*
- *We have successfully construct a Lyapunov function v_5 with $N_{final} = 96$ grid points, see figure 4.11. Moreover, the function v_5 has negative orbital derivative on a checking grid $X_{check} \subset K$, with $h_{check} = 10^{-2}$.*

For this example, when the refinement algorithm failed to construct a Lyapunov function with 60 grid points, the first modified algorithm did so with 96 points. Table 4.3, presents



(a) The Delaunay triangulation of X_4. (b) The 4 clusters and their centres.

Figure 4.10: (a) The Delaunay triangulation for X_4, where the vertices of the triangles are our 93 grid points (blue *), (b) the points in PO_3 (green *) located in the areas where the orbital derivative is positive and grouped into 4 clusters based on the triangles they belong to. The black (*) represent the centres of the clusters calculated by the k-means function.



(a) The final set of grid points. (b) The Lyapunov function v_5.

Figure 4.11: (a) The final set of grid points $X_5 = 96$ points. As we see, there are no patches where $v_5'(x, y) > 0$, (b) the Lyapunov function v_5 constructed with the first modified grid refinement algorithm.

the performance of the modified algorithm in brief.

	Steps performed in order	No. of points added	N	Time (whole process)
A1	refinement	12	60	25.6 sec.
B1	extension	8	68	3 min. 28 sec.
A2	refinement	0	68	0.5 sec.
B2	extension	8	76	5 min. 15 sec.
A3	refinement	16	92	1 min. 16 sec.
B3	extension	4	96	5 min. 58 sec.
				total= 16 min. 23 sec.

Table 4.3: The steps of the first modified grid refinement algorithm performed in sequence, with the number of points added after each step as well as the total number of grid points N. The table also shows the whole time needed for each process, including the steps of the algorithm + solving the linear system for the obtained set of grid points + plotting the orbital derivative of the corresponding RBF approximants.

Remark 4.1. *Due to the geometrical complexity of the Delaunay triangulation in high dimensions, performing the extension process (steps 4-7) in the first modified algorithm turns out to be hard and needs a long time. Therefore, we did not apply the algorithm to the 3-D system.*

4.2.3 The 2nd modified algorithm: using subtractive clustering

The extension step of the second modified grid refinement algorithm uses the subtractive clustering technique to find the cluster (patches) centres. This task will be done by using the *"subclust"* MATLAB function, a simple and effective tool which determines the cluster centres of any d-dimensional data set in a single run.

The Algorithm Strategy:

- A1: A refinement process

  Again, consider the last constructed function v_n, before the termination of the refinement algorithm, with the grid points X_n, such that v_n still has positive orbital derivative in some areas. Then

- B1: An extension process

1. Fix a test grid $X_{test} \in K \subset \mathbb{R}^d$ with a specified density h_{test}, to check the sign of the orbital derivative of v_n at each point of X_{test}.
2. Build an array PO containing all the points $x \in X_{check}$ with positive orbital derivative, i.e., $PO = \{x \in X_{test} \mid v_n'(x) > 0\}$, excluding a small neighbourhood of the equilibrium point.
3. Apply the subclust MATLAB function to PO with radius=R, which finds the clusters centres by the subtractive clustering technique, and returns them in a set C. Note that, if no centres returned, we choose a smaller radius.
4. Add the obtained centres to our existing grid points and define a new grid $X_{n+1} = X_n \cup C$.
5. Calculate the function v_{n+1} using the RBF method with X_{n+1}.
   If $v_{n+1}' < 0$, STOP
   else

- A2: A refinement process

  If v_{n+1} still has areas where $v_{n+1}' > 0$, then we perform steps 3-6 of the refinement algorithm, introduced in Section 3.1.2, to check if it adds more points and continue until it terminates again. If v_{final}, the function constructed with the final set of grid points X_{final} obtained after the termination of the current refinement step, still has positive orbital derivative at some points in K, then

- B2: An extension process

  We apply the extension algorithm again (steps 1-5).

Again, iterate this process until we successfully construct a Lyapunov function on K.

4.2.4 Numerical Examples

For the sake of comparing the performance and the output of both algorithms, we will consider the same cases solved with the previous algorithm. Note that, for all our examples we use $R = 0.2$.

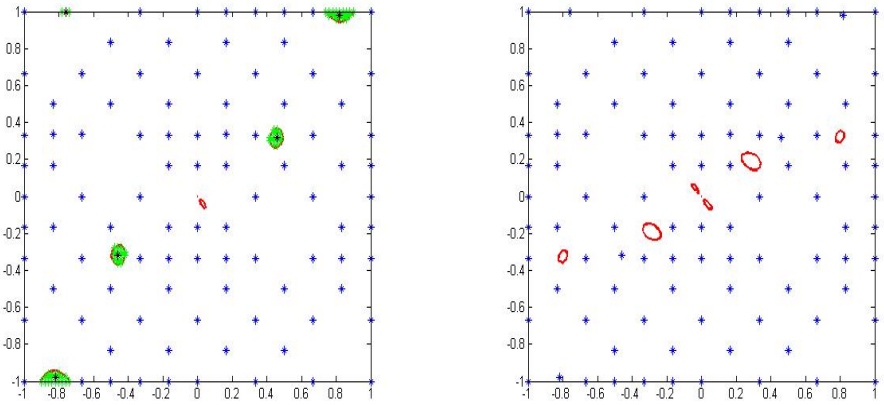
Example 4.3. *We will solve the same case considered in Example 4.1.*

1. *A1: Refinement process:*

- *The refinement started with* $N_{initial} = 16$ *regular grid points and terminated after 9 refinement steps at* $X_9 = 93$ *points.*

2. *B1: 1st extension process*

- *Choose a grid* X_{test} *with* $h_{test} = 0.02$ *to check the sign of the orbital derivative of the function* $v_9(x, y)$ *over all the points* $(x, y) \in X_{test}$*, which will in turn produce the array* PO_1 *of* 74 *points with positive orbital derivative (green *) in figure 4.12 (a).*
- *Run the subclust MATLAB function on* PO_1 *with radius* $R = 0.2$ *which returns a matrix* $C_1 = \{5\}$ *cluster centres, (black *) in figure 4.12 (a).*
- *Then, we add the 5 centres to our existing grid points* X_9 *and obtain a new set of points* $X_{10} = X_9 \cup C_1 = \{x_i\}_{i=1}^{98}$.
- *Calculate the function* v_{10} *with* X_{10}*. The level set of* $v'_{10}(x, y) = 0$ *in figure 4.12 (b), shows that there are still areas where* $v'_{10} > 0$*. Thus, we apply the refinement algorithm with* X_{10}*, to check if it adds more points.*



(a) The first extension step.

(b) The level set of $v'_{10}(x, y) = 0$.

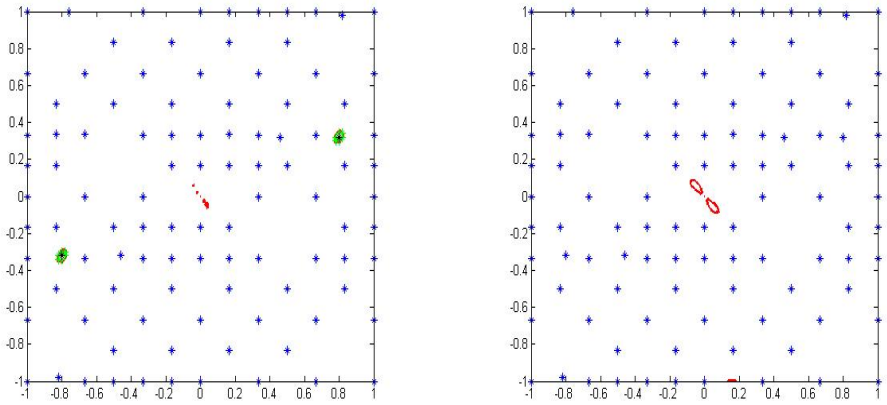
Figure 4.12: (a) The 74 points in PO_1 (green *) lie in the patches where $v'_9(x, y) > 0$, and the centres to be added to X_9 (black *), (b) the level set of $v'_{10}(x, y) = 0$, where the red areas are the patches remaining where $v'_{10}(x, y) > 0$.

3. *A2: Refinement process:*

- *The refinement algorithm adds two more points the grid points giving the set* $X_{11} = X_{10} \cup \{y_j\}_{j=1}^{2} = \{x_i\}_{i=1}^{100}$*, then it terminates again.*
- *The function* v_{11} *still has positive orbital derivative in some patches. Therefore, another extension step need to be performed.*

4. *B2: 2nd extension process:*

- *Checking the sign of the orbital derivative of the function* $v_{11}(x, y)$ *over the grid* X_{test} *gives a new array* PO_2 *of* 14 *points with positive orbital derivative.*
- *Again, we apply the subclust function on* PO_2 *with* $R = 0.2$*, which gives two centres* $C_2 = 2$*, (black *) in figure 4.13 (a).*
- *Adding the 2 centres to the set* X_{11} *gives a new set of grid points* $X_{12} = X_{11} \cup C_2 = \{x_i\}_{i=1}^{102}$.
- *We calculate the function* v_{12} *with* X_{12}*, which still has patches where the orbital derivative is positive, see figure 4.13 (b).*



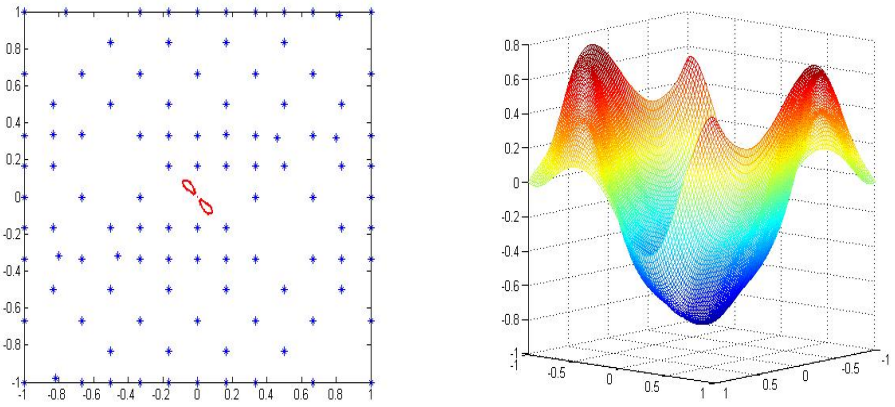
(a) The second extension step.

(b) The level set of $v_{12}(x, y) = 0$.

Figure 4.13: (a) The level set of $v'_{11}(x, y) = 0$, the grid points $X_{11} = 100$ (blue *) and the 14 points in PO_2 (green *) filling the patches where $v_{11}(x, y) > 0$. We can see the 2 cluster centres to be added (black *), (b) the level set of $v'_{12}(x, y) = 0$ with the patches where the orbital derivative is positive.

5. *A3: Refinement process*

- *Performing one more step of the refinement algorithm adds two points to the grid and gives a new set* $X_{13} = X_{12} \cup \{y_j\}_{j=1}^{2} = \{x_i\}_{i=1}^{104}$.
- *Finally, the function* v_{13} *which is constructed with* $X_{13} = 104$ *grid points has no more patches with positive orbital derivative except for a small excluded neighbour hood* $E_{nh} = [-0.1, 0.1]^2$*, of the equilibrium* $(0, 0)$*. See figure 4.14 (a). Moreover, the function* v_{12} *is a Lyapunov function since it has negative orbital derivative on a checking grid* X_{check} *with* $h_{check} = 10^{-2}$.



(a) The level set of $v'_{13}(x, y) = 0$

(b) The constructed Lyapunov function $v_{13}(x, y)$.

Figure 4.14: (a) The level set of $v'_{13}(x, y) = 0$ and the grid points $X_{13} = 104$ (blue *), as we can see there are no patches remaining at the end, (b) the constructed Lyapunov function $v_{13}(x, y)$ with the second extension algorithm.

The performance of the second modified algorithm to solve this case is summarized in Table 4.4.

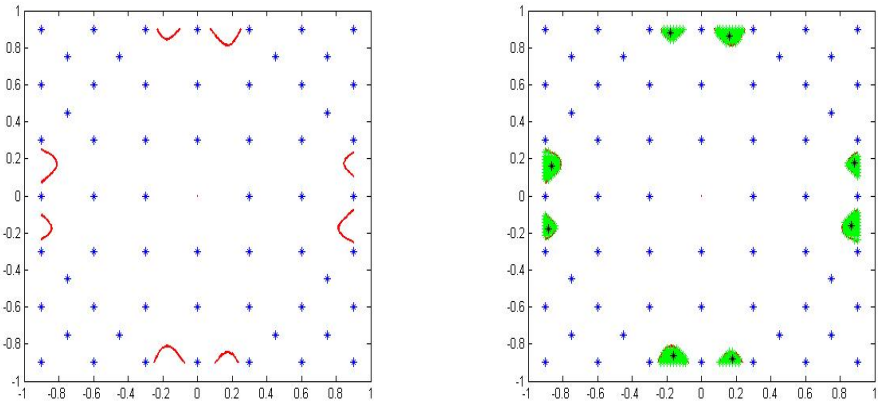
	Steps preformed in order	No. of points added	N	Time (whole process)
A1	refinement	77	93	45.66 sec.
B1	extension	5	98	2 min. 34 sec.
A2	refinement	2	100	47.05 sec.
B2	extension	2	102	2 min. 54 sec.
A3	refinement	2	104	48.35 sec.
				total= 7 min. 49 sec.

Table 4.4: The steps of the second modified grid refinement algorithm performed in sequence, with the number of points added after each step as well as the total number of grid points N. The table also shows the whole time needed for each process, including the steps of the algorithm + solving the linear system for the obtained set of grid points + plotting the orbital derivative of the corresponding RBF approximants.

Comparing the results of Table 4.4 to the the results of the first modified algorithm, i.e., Table 4.2 shows that although we needed the same number of grid points ($N_{final} = 104$ points) to construct a Lyapunov function with both algorithms, the second algorithm did so with a total of 7 min. and 49 sec. while the first algorithm needed a total of 9 min.

Example 4.4. *Consider the same case solved in Example 4.2.*

1. *A1: Refinement process*
   - *The refinement algorithm started with a regular grid $N_{initial} = 48$, then it terminated after one refinement step at $X_1 = 60$ grid points, see Figure 4.15 (a).*
2. *B1: 1st extension process*
   - *Fix a test grid X_{test} with $h_{test} = 0.018$, to check the sign of the orbital derivative of v_1, which will in turn build the array $PO_1 = \{(x, y) \in X_{test} \subset \mathbb{R}^2 \mid v_1'(x, y) > 0\}$. The green (*) in Figure 4.15 (b).*
   - *Running the subclust MATLAB function on PO_1 with radius $R = 0.2$, returns a matrix $C_1 = 8$ cluster centres, black (*) in Figure 4.15 (b).*
   - *We add the 8 centres to the previous grid points X_1 to obtain a new set of grid points $X_2 = X_1 \cup C_1 = \{x_i\}_{i=1}^{68}$.*



(a) The level set of $v_1'(x, y) = 0$ (b) The first extension step.

Figure 4.15: (a) The level set of $v_1'(x, y) = 0$ of the constructed function v_1 after the last refinement step, started with 48 points and ended up with 60 points, where the (red) patches are the areas where $v_1'(x, y) > 0$. (b) The Delaunay triangulation for X_1, the points in PO_1 (green *) grouped into 8 clusters, and the centres to be added to X_1 (black *).

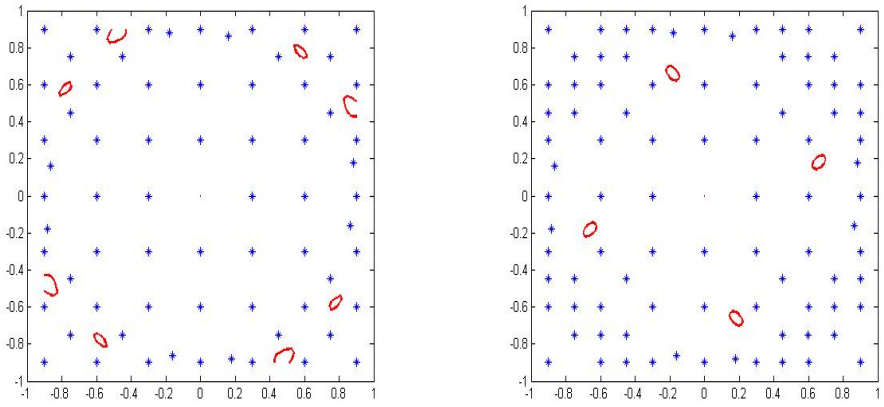
- *Calculate the RBF function v_2 with X_2. However, the level set of $v_2'(x, y) = 0$ shows that v_2 has some areas where the orbital derivative is positive. See Figure 4.16 (a).*

3. *A2: Refinement process*

- *3 refinement steps add 20 points to our existing grid points X_2, giving a new set $X_3 = X_2 \cup \{y_j\}_{j=1}^{20} = \{x_i\}_{i=1}^{88}$, then it terminates again.*
- *As shown in Figure 4.16 (b), the constructed RBF function v_3 with X_3 also has areas where the orbital derivative is positive.*

4. *B2: 2nd extension process*

- *Produce a new array PO_2, containing all the the points $(x, y) \in X_{test}$ where $v_3'(x, y) > 0$.*
- *We apply the subclust MATLAB function on PO_2 with radius $R = 0.2$ and obtain a new set of cluster centres $C_2 = 4$ centres, the black (*) in Figure 4.17 (a).*



(a) The level set of $v_2'(x, y) = 0$ (b) The level set of $v_3'(x, y) = 0$

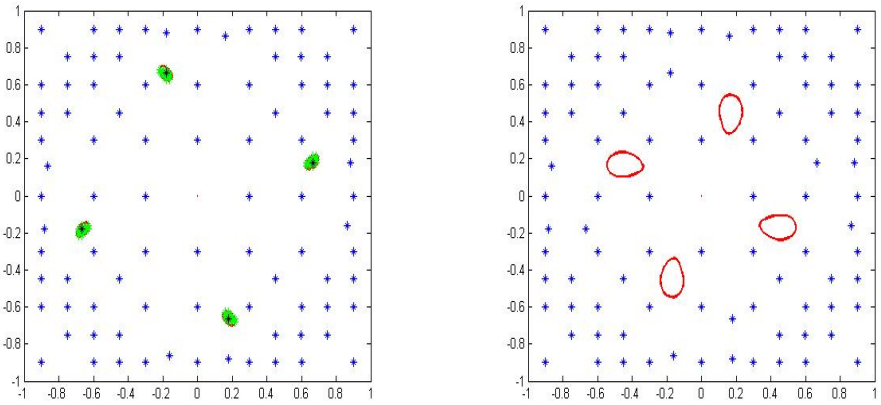
Figure 4.16: (a) The level set of $v_2'(x, y) = 0$ of the constructed function v_2 with the set X_2 obtained from the first extension step, where the (red) patches are the areas where $v_2'(x, y) > 0$. (b) The level set of $v_3'(x, y) = 0$ of the constructed function v_3 with the set of grid points obtained from the second refinement step X_3. Again, the (red) patches are the areas where $v_3'(x, y) > 0$

- *Add the 4 centres to the set X_3 and get a new set of grid points $X_4 = X_3 \cup C_2 = \{x_i\}_{i=1}^{92}$. Then, use the set X_4 to construct a function v_4. However the function still has areas where the orbital derivative of v_4 is positive. See Figure 4.17 (b).*

5. *A3: Refinement process*

- *Performing one refinement step adds 4 grid points to X_4, giving the set $X_5 = X_4 \cup \{y_j\}_{j=1}^{4} = \{x_i\}_{i=1}^{96}$.*
- *The constructed RBF function v_5 with X_5 has no more patches with positive orbital derivative, as shown in Figure 4.18 (a). Moreover, the function v_5 has negative orbital derivative on a checking grid X_{check} with $h_{check} = 10^{-2}$, thus it is a Lyapunov function, see Figure 4.18 (b).*

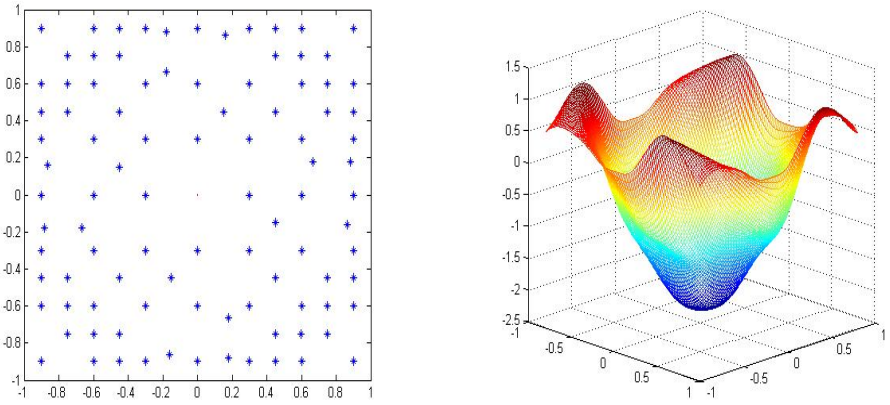
From Table 4.3 and Table 4.5, we can see a significant reduction in the time needed to construct a Lyapunov function with the second modified algorithm as it needed a total of 5 min. and 44 sec., while the first algorithm needed a total of 16 min. and 23 sec. Again,



(a) The second extension step.

(b) The level set of $v_4'(x, y) = 0$.

Figure 4.17: (a) The points in PO_2 (green *) grouped into 4 clusters, and the cluster centres (black *), calculated by the subtractive clustering method, (b)the level set of $v_4'(x, y) = 0$ of the constructed function v_4, where the (red) patches are the areas where $v_4'(x, y) > 0$.



(a) The level set of $v_5'(x, y) = 0$

(b) The constructed Lyapunov function $v_5(x, y)$.

Figure 4.18: (a) The level set of $v_5'(x, y) = 0$ and the grid points $X_5 = 96$ (blue *), as we can see there are no patches remaining at the end, (b) the constructed Lyapunov function $v_5(x, y)$ with the second modified grid refinement algorithm.

the required number of grid points still the same $N_{final} = 96$ *points.*

	Steps preformed in order	No. of points added	N	Time (whole process)
A1	refinement	12	60	25.6 sec.
B1	extension	8	68	1 min. 34 sec.
A2	refinement	20	88	40 sec.
B2	extension	4	92	2 min. 3 sec.
A3	refinement	4	96	42 sec.
				total= 5 min. 44 sec.

Table 4.5: The steps of the second modified grid refinement algorithm performed in sequence, with the number of points added after each step as well as the total number of grid points N. The table also shows the whole time needed for each process, including the steps of the algorithm + solving the linear system for the obtained set of grid points + plotting the orbital derivative of the corresponding RBF approximants.

Example 4.5. *Consider the three-dimensional Example 3.4, in Table 3.3 we have investigated the influence of the starting grid on the refinement algorithm. As we can see, there are some cases where the refinement terminates with patches remaining at the end.*

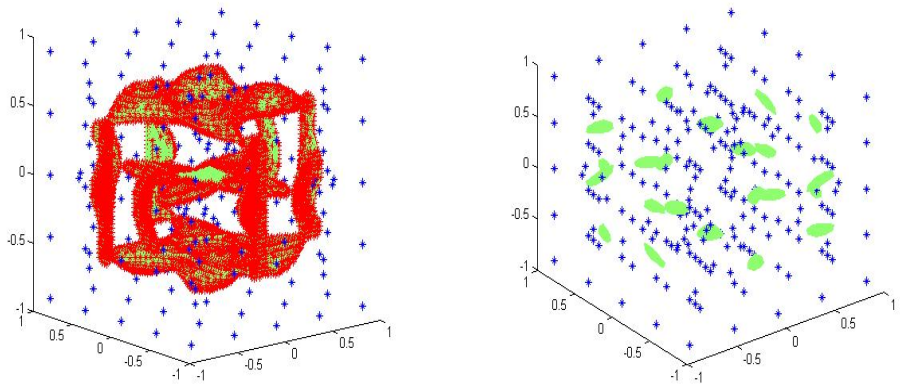
1. *A1: Refinement process*

   - *We started the refinement with* $N_{initial} = 124$ *regular points and ended up with 180 points after two refinement steps, see Figure 3.15 (a).*

2. *B1: 1st extension process*

   - *We choose a grid* $X_{test} = \{(x, y, z) \in \mathbb{R}^3 \mid x, y, z \in \{0, \pm h_{test}, \ldots, \pm 0.9\}\} \backslash E_{nh}$, *where* $E_{nh} = [-0.2, 0.2]^3$ *and* $h_{test} = 0.03$ *to check the sign of the orbital derivative of* $v_3(x, y, z)$ *over all the points* $(x, y, z) \in X_{test}$. *This will produce the array* PO_1 *of* 13908 *points where* $v_3'(x, y, z) > 0$, *(red *) in Figure 4.19 (a).*
   - *Passing the array* PO_1 *into the subclust MATLAB function with* $R = 0.2$, *returns a matrix* C_1 *with* 36 *cluster centres.*
   - *We add the 36 centres into our existing grid points and create a new set* $X_4 = X_3 \cup C_1 = \{x_i\}_{i=1}^{216}$.

- *Calculate the function $v_4(x, y, z)$ with X_4, which still has areas with positive orbital derivative, see Figure 4.19 (b).*



(a) The first extension step. (b) The level set of $v_4'(x, y, z) = 0$.

Figure 4.19: (a) The level set of $v_3'(x, y, z) = 0$ (green area), the 13908 points in PO_1 where $v_3'(x, y, z) > 0$ (red *) and we can see some of the cluster centres as (black *), (b) the level set of $v_4'(x, y, z) = 0$ after adding the 36 cluster centres to the grid points, where the green patches are the patches where $v_4'(x, y, z) > 0$.

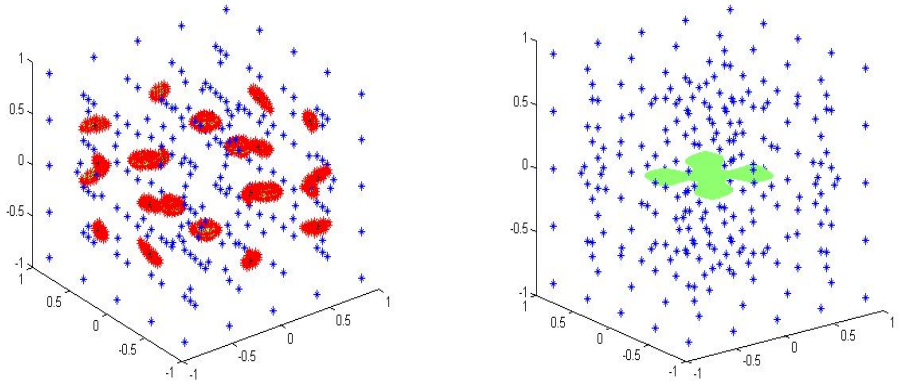
3. *A2: Refinement process*

   - *The refinement algorithm with X_4 did not add any more points.*

4. *B2: 2nd extension process*

   - *We check the sign of the orbital derivative of $v_4'(x, y, z)$ over X_{test} and obtain the array PO_2 of 1148 points with positive orbital derivative.*
   - *Applying the subclust function to PO_2 with $R = 0.2$ gives $C_2 = 22$ cluster centres, (black *) in Figure 4.20 (a).*
   - *Add the 22 centres to the existing set of grid points and we get a new set $X_5 = X_4 \cup C_2 = \{x_i\}_{i=1}^{238}$.*
   - *The function $v_5(x, y, z)$ calculated with X_5 still has areas where the orbital derivative is positive, as shown in Figure 4.20 (b).*

5. *A3: refinement process*



(a) The second extension step. (b) The level set of $v_5'(x, y, z) = 0$.

Figure 4.20: (a) The 1148 points in PO_2 lie inside the patches where $v_4'(x, y, z) > 0$ (red *) and cluster centres are shown in (black *), (b) the level set of $v_5'(x, y, z) = 0$ after adding the 22 cluster centres to the grid points, where the green patch is the patch where $v_5'(x, y, z) > 0$.

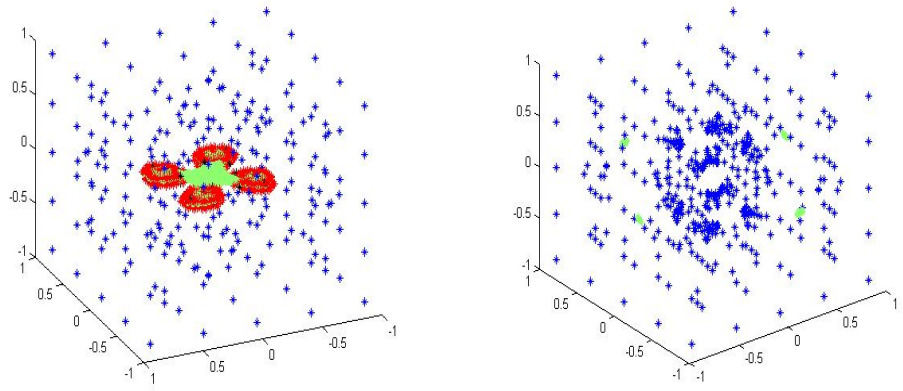
- *The refinement with X_5 did not add any points.*

6. *B3: 3rd extension process*

- *Following the same steps gives the array PO_3 with 952 points where $v_5'(x, y, z) > 0$, (red *) in Figure 4.21 (a).*
- *The subclust function with $R = 0.2$ returns $C_3 = 44$ cluster centres, where can be seen in Figure 4.21 (a).*
- *Add the 44 centres to X_5, then we get a new set of grid points $X_6 = X_5 \cup C_3 = \{x_i\}_{i=1}^{282}$.*
- *Calculate the function v_6 with X_6. This function still has some patches where the orbital derivative is positive.*

7. *A4: Refinement process*

- *The refinement algorithm with X_6 adds a 168 points to the grid in only one step, before it terminates again, giving the new set $X_7 = X_6 \cup \{y_j\}_{j=1}^{168} = \{x_i\}_{i=1}^{450}$.*



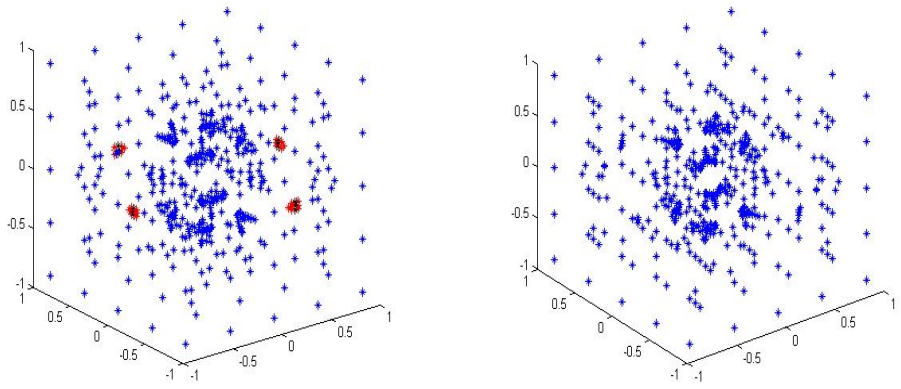
(a) The Third extension step. (b) The refinement with $X_6 = 282$ points.

Figure 4.21: (a) The figure shows where the 952 points in PO_3 were placed (red *) where the green area in the middle is the excluded neighbourhood E_{nh} around the equilibrium $(0, 0, 0)$, and also shows some of the cluster centres (black *). (b) The level set $v_7'(x, y, z) = 0$ of the orbital derivative after the refinement step started with with $X_6 = 282$ and ended up adding 163 points to the grid, where the small patches (green areas) remaining still have positive orbital derivative.

- *The function $v_7(x, y, z)$ calculated with X_7 has positive orbital derivative in some areas, see Figure 4.21 (b).*

8. *B4: 4th extension process*

   - *Calculate the array PO_4 with 40 points where $v_7'(x, y, z) > 0$, (red *) in Figure 4.22 (a).*
   - *Passing PO_4 into the subclust function with $R = 0.2$ returns $C_4 = 12$ cluster centres.*
   - *By adding the 12 centres to the previous set of grid points we obtain a new one $X_8 = X_7 \cup C_4 = \{x_i\}_{i=1}^{462}$.*
   - *Finally, with $X_8 = 462$ grid points, we were able to construct a Lyapunov function $v_8(x, y, z)$ which has negative orbital derivative on a checking grid X_{check} with $h_{check} = 0.018$, see Figure 4.22 (b).*



(a) The final extension step. (b) The final set of grid points $X_8 = 462$.

Figure 4.22: (a) The 40 points in PO_4, (red *), lie inside the patches where $v_7'(x, y, z) > 0$ and some of the cluster centres are shown as (black *), (b) the level set $v_8'(x, y, z) = 0$ of the orbital derivative calculated with $X_8 = 462$ grid points, as we can see there are no patches remaining at the end.

	Steps preformed in order	No. of points added	N	Time (whole process)
A1	refinement	56	180	30 min. 30 sec.
B1	extension	36	216	1 hr. 5 min.
A2	refinement	0	216	9.9 sec.
B2	extension	22	238	1 hr. 13 min. 7 sec.
A3	refinement	0	238	12.4 sec.
B3	extension	44	282	1 hr. 18 min. 24 sec.
A4	refinement	168	450	10 min. 6 sec.
B4	extension	12	462	2 hr. 40 min.
				total= 6 hr. 57 min. 29 sec.

Table 4.6: The steps of the second modified grid refinement algorithm performed in sequence, with the number of points added after each step as well as the total number of grid points N. The table also shows the whole time needed for each process, including the steps of the algorithm + solving the linear system for the obtained set of grid points + plotting the orbital derivative of the corresponding RBF approximants.

To sum up, the modified grid refinement algorithms have managed to construct a Lyapunov function successfully, with the same number of grid points, when the original refinement algorithm terminated before constructing one. Determining the cluster centres with the second modified algorithm (using the subtractive clustering) is relatively easy and the whole process requires less computation time than the first one (using the k-means clustering). Moreover, it is preferable when dealing with high dimensional systems as the *subclust* MATLAB function needs only one step to determine the cluster centres for a data set of arbitrary dimension. On the other hand, the Delaunay triangulation becomes geometrically complicated for high-dimensional data sets, which, in turn, makes finding the cluster centres much harder with the first modified algorithm.

For both modified algorithms, we could have skipped the refinement procedures and just use the new extension procedures to refine the whole grid instead of placing points only in specific patches. However, the process of determining the cluster centres is not straightforward and needs more steps and time, which makes it an undesirable method to refine the grid with compared to the refinement described in Chapter 3. Therefore, this refinement technique (using cluster centres) will be used only when the original refinement algorithm fails to construct a Lyapunov function.

Finally, constructing Lyapunov functions with either the original refinement algorithm, described in Chapter 3, or with the modified refinement algorithms, described in Chapter 4, leads to a similar distribution of grid points in the compact set K with no big variation of the required number of points.

Chapter 5

The Verification Estimates

5.1 General Formulation of the Verification Estimates

The constructed function v using the RBF method on either a regular grid or by the refinement algorithms, cannot be guaranteed to have negative orbital derivative in the whole set K as there might be points in between the grid points where the orbital derivative is positive. According to the theoretical error estimate (2.12), the approximant v is a Lyapunov function (i.e., $v'(x) < 0, \forall x \in K$) if the RBF grid is fine enough. However, we can not tell in advance how dense the grid should be (i.e. how small the fill distance h should be), since this error estimate depends on unknown quantities such as V. So far, in numerical examples, we have checked the sign of v' on a fine checking grid X_{check}. However, we need a more reliable verification of the negativity of the orbital derivative of a constructed function v on a compact set K.

Therefore, to solve this problem, we have come up with two verification estimates that are based on a Taylor approximation and rely on the first and second derivatives of the orbital derivative. For the derivation of these estimates we make use of the special form of the RBF approximant and its orbital derivative (2.8) and (2.9), respectively. The other main ingredient of these verification estimates is a checking grid Y_{od}, where we actually use these estimates to determine the density of Y_{od}. This checking grid is then used to verify the negativity of the orbital derivative over a different grid than the RBF one. The strategy of using these estimates will be as follows:

- Use the estimates to obtain the required density of the checking grid Y_{od}.

- Calculate the orbital derivative at all points of Y_{od}.
- If $v'(y)+$an error term < 0, $\forall y \in Y_{od}$, then the constructed function v is a Lyapunov function otherwise we might need a finer RBF grid to construct the function v or a finer checking grid Y_{od}.

The advantage of these verification estimates is that they depend on known and easily calculated quantities as we will see in the coming sections.

Let $K \subset \mathbb{R}^d$ be a fixed compact set, and let

1. $X_{RBF} = \{x_1, x_2, \ldots, x_N\} \subset K$, used for the calculation of the approximant v of the solution of one of the two Lyapunov functions satisfying $V' = -\|x-x_0\|^2$ or $V' = -\bar{c}$, using the radial basis functions approximation method. Because of the ansatz of these Lyapunov functions, we guarantee that the orbital derivative is negative at each grid point in X_{RBF}, i.e., $v'(x_i) < 0$ for all $x_i \in X_{RBF}$, $i = 1, \ldots, N$.
2. $Y_{od} = \{y_1, y_2, \ldots, y_M\} \subset K$, used to check the sign and value of the orbital derivative of the constructed Lyapunov function on a different (usually finer but not necessarily) grid than X_{RBF}.

Note that the X_{RBF} and Y_{od} are subsets of K, however X_{RBF} is not required to be subset of Y_{od}.

5.1.1 First Estimate: Using the First Derivative of the Orbital Derivative.

We derive the first verification estimate by satisfying the Lipschitz continuity condition for the orbital derivative of the constructed function v, at every point $x \in K$. The Lipschitz condition is an upper bound on how the function changes for every pair of points x and all y sufficiently near to x. Thus, satisfying the Lipschitz condition for the orbital derivative v' at every point $x \in K$ and all $y \in Y_{od} \subset K$, sufficiently near to x, will lead to an estimate in terms of the first derivative of the orbital derivative and the fill distance $h_{\|.\|_2} \geq \|x-y\|_2$.

Theorem 5.1. *[48, Theorem 3.4.1] (Mean value theorem for functions of several variables) Suppose $\eta : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable function on the open convex set*

Ω. Let $a, b \in \Omega$ and $\gamma(t) = a + t(b-a),\ t \in [0,1]$ be the line segment joining a and b. Then there exists ϑ on $\gamma(t)$ such that

$$\eta(a) - \eta(b) = \langle \nabla\eta(\vartheta), a - b\rangle.$$

One of the important corollaries of the mean value theorem is what is called a Lipschitz condition.

Corollary 5.1. *[48, Corollary 3.4.2] (Lipschitz continuity condition) Let $\eta : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ be a differentiable function on a convex subset K of Ω. If $\|\nabla\eta(z)\|_2 \leq M^*$ for all $z \in K$, then η is Lipschitz continuous in K, i.e.*

$$|\eta(x) - \eta(y)| \leq M^* \|x - y\|_2 \tag{5.1}$$

for all $x,\ y \in K$.

For our estimate, we set $\eta = v'$ in (5.1), where $v' : K \subset \mathbb{R}^d \to \mathbb{R}$ is the orbital derivative of an approximant v, and for all $x \in K$ we choose the nearest $y \in Y_{od}$ such that $\|x - y\|_2 \leq h_{\|.\|_2}$. Then, the general formulation of the first estimate will be

$$\begin{aligned} |v'(x) - v'(y)| &\leq \max_{z \in co(K)} \|\nabla v'(z)\|_2 \, \|x - y\|_2 \\ \Rightarrow v'(x) &\leq \max_{y \in Y_{od}} v'(y) + \max_{z \in co(K)} \|\nabla v'(z)\|_2 \, h_{\|.\|_2} \end{aligned} \tag{5.2}$$

for all $x \in K$ and $y \in Y_{od}$, and where $h_{\|.\|_2}$ is the fill distance defined as

$$h_{\|.\|_2} = \max_{x \in K} \min_{y \in Y_{od}} \|x - y\|_2 \tag{5.3}$$

5.1.2 Second Estimate: Using the Second Derivative of the Orbital Derivative.

For the purpose of deriving the formula of this estimate, we need to recall an existing proposition presented in [4], which requires suitable triangulations of the set $K \subset \mathbb{R}^d$. Therefore, we start by defining a k-simplex, and triangulations, and then state the original version of the proposition.

Let $x_0, \ldots, x_k$ be affinely independent vectors in $\mathbb{R}^d$, that is $\sum_{i=1}^k \lambda_i (x_i - x_0) = 0$, implies that $\lambda_i = 0$ for $i = 1, \ldots, k$. The convex combinations of these vectors $x_0, \ldots, x_k$ defines a k-simplex $\mathcal{T} := co\{x_0, \ldots, x_k\} = \{\sum_{i=0}^k \lambda_i x_i : 0 \leq \lambda_i \leq 1, \sum_{i=0}^k \lambda_i = 1\}$.

Definition 5.1. *(Triangulation). Let* $\mathrm{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$ *be finitely many* k*-simplices. Then* T *is a triangulation of the set* $K \subset \mathbb{R}^d$ *if* $K = \bigcup_{i=1}^{m} \mathcal{T}_i$*, and for every* $\mathcal{T}_i, \mathcal{T}_j \in \mathrm{T}$*,* $i \neq j$*, then either* $\mathcal{T}_i \cap \mathcal{T}_j = \emptyset$*, or* $\mathcal{T}_i \cap \mathcal{T}_j =$ *a common face.*

For an example of such a triangulation, see Figure 5.1.

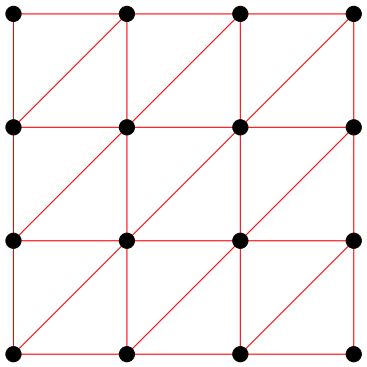


Figure 5.1: A suitable triangulation in $\mathbb{R}^2$

Now, we present the original proposition, which is the base of our estimate.

Proposition 5.1 (4.1 in [4])**.** *Let* $(x_0, x_1, \ldots, x_k) \in \mathbb{R}^d$ *be affinely independent vectors, define* $\mathcal{T} := co\{x_0, x_1, \ldots, x_k\}$*,* $h :=$*diam*$(\mathcal{T})$ *and consider a convex combination* $\sum_{i=0}^{k} \lambda_i x_i \in \mathcal{T}$*. Let* $U \subset \mathbb{R}^d$ *be an open set with* $\mathcal{T} \subset U$*.*
If $w \in C^2(U, \mathbb{R})$*, then*

$$\left| w\left(\sum_{i=0}^{k} \lambda_i x_i\right) - \sum_{i=0}^{k} \lambda_i w(x_i) \right| \leq B_H h^2.$$

where $B_H := \max_{z \in \mathcal{T}} \|H(z)\|_2$ *and* $H(z)$ *is the Hessian of* w *at* z*.*

This proposition shows that the difference between the function values $w(x)$ at a point $x \in \mathcal{T}$, written as convex combination of the vertices of $\mathcal{T}$, and the convex combination of the function values $w(x_i)$ at the vertices, is bounded by some estimates on the Hessian of w and the diameter h of $\mathcal{T}$, which is the largest $\|.\|_2$ distance between any two vertices in $\mathcal{T}$. Thus, the error is small if the simplex is small.

We formulate our estimate by replacing the function w by the orbital derivative v'

$$\left| v'\Big(\sum_{i=0}^{k} \lambda_i x_i\Big) - \sum_{i=0}^{k} \lambda_i v'(x_i) \right| \leq B_H \; h^2,$$

thus, the general formulation of the second estimate will look like

$$v'\left(\sum_{i=0}^{k}\lambda_i x_i\right) \leq \max_{y\in Y_{od}} v'(y) + \max_{z\in\mathcal{T}} \|H(z)\|_2 \; h^2_{\|.\|_2}. \tag{5.4}$$

Remark 5.1. *In both estimates (5.2) and (5.4), we need to estimate the maximal value of* $\|\nabla v'(z)\|_\infty$ *and* $\|H(z)\|_{\max}$ *rather than the maximum of the* L_2*-norm of both quantities, where* $\|.\|_{\max}$ *is the maximum modulus of all elements of a square matrix* $A \in \mathbb{R}^{d\times d}$*, i.e.,*

$$\|A\|_{\max} = \max_{1\leq i,j\leq d} |a_{i,j}|.$$

It turns out that we actually will obtain a bound on $\|\nabla v'(z)\|_\infty$ *and* $\|H(z)\|_{\max}$*. Consequently*

1. *The first estimate (5.2) becomes*

$$v'(x) \leq \max_{y\in Y_{od}} v'(y) + \sqrt{d}\;\left(\max_{x\in co(K)} \max_{j=1,\ldots,d} \left|\frac{\partial v'(x)}{\partial x_j}\right|\right)\; h_{\|.\|_2}$$

   where we use $\max\limits_{z\in co(K)} \|\nabla v'(z)\|_2 \leq \sqrt{d} \max\limits_{z\in co(K)} \|\nabla v'(z)\|_\infty$*,*
   and $\frac{\partial v'(x)}{\partial x_j}$ *is the first derivative of the orbital derivative.*

2. *The second derivative (5.4) becomes*

$$v'\Big(\sum_{i=0}^{k}\lambda_i x_i\Big) \leq \max_{y\in Y_{od}} v'(y) + d\;\left(\max_{x\in\mathcal{T}} \max_{i,j=1,\ldots,d} \left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right|\right)\; h^2_{\|.\|_2}.$$

   where we use $\max\limits_{z\in\mathcal{T}} \|H(z)\|_2 \leq d \max\limits_{z\in\mathcal{T}} \|H(z)\|_{\max}$*,*
   and $\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}$ *is the second derivative of the orbital derivative.*

Note that, the L_2 *and the* $L_{\max}$ *matrix norms are defined in the List of symbols. Obviously, the estimates can be improved by changing the* L_2*-norm into the* $L_\infty\backslash L_{\max}$*-norm. This issue will be investigated in detail later in this section but first we will define the first and second derivatives of the orbital derivative.*

5.1.3 The first and second derivatives of the orbital derivative

Theorem 5.2 (The first derivative of the orbital derivative)**.** *Let* $v \in C^2(\mathbb{R}^d, \mathbb{R})$ *be an RBF approximant as in (2.8) with a radial basis function* $\psi := \psi_0(r) \in C^3$*. Let* K *be a*

compact set and $\{x_1, \dots, x_N\} \subseteq K$ *be a set of* N *pairwise distinct points which are no equilibria. Then the first derivative of the orbital derivative* v' *can be bounded by*

$$\left|\frac{\partial v'(x)}{\partial x_j}\right| \leq \boldsymbol{\beta}\Big(\Psi_{3,3}\, F^2 + 3\,\Psi_{2,1}\, F^2 + \Psi_{2,2}\, F\, D_1 + \Psi_{1,0}\, F\, D_1\Big), \quad \forall\, x \in K, \tag{5.5}$$

where :

- $\Psi_{i,k} := \sup\limits_{r\in[0,\infty)} |\psi_i(r)| \cdot r^k,\ i = 1,2,3$ *and* $k \in \mathbb{N}_0^+$.
- *The functions* $\psi_i(r)$ *are defined as*

$$\psi_i(r) = \frac{\frac{d}{dr}\,\psi_{i-1}(r)}{r}, \quad for \quad r > 0. \tag{5.6}$$

  and can be continuously extended to $r = 0$.

- $F := \max\limits_{x\in K} \|f(x)\|_2$.
- $D_1 := \max\limits_{x\in K} \max\limits_{j\in 1,\dots,d} \|\frac{\partial f(x)}{\partial x_j}\|_2$.
- $\boldsymbol{\beta} := \sum_{k=1}^{N} |\beta_k|$.

Proof. Recall the formula of the orbital derivative (2.9)

$$v'(x) = \sum_{k=1}^{N} \beta_k \big[\psi_2(\|x - x_k\|_2)\langle x - x_k, f(x)\rangle\langle x_k - x, f(x_k)\rangle - \psi_1(\|x - x_k\|_2)\langle f(x), f(x_k)\rangle\big]$$

By differentiating v' we get

$$\frac{\partial v'(x)}{\partial x_j} = \sum_{k=1}^{N} \beta_k \left(\frac{\psi_2'(\|x-x_k\|_2)}{\|x-x_k\|_2} (x-x_k)_j \langle x-x_k, f(x)\rangle \langle x_k-x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) f_j(x) \right.$$
$$\langle x_k-x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) \left(\sum_{l=1}^{d} (x-x_k)_l \frac{\partial f_l(x)}{\partial x_j} \right) \langle x_k-x, f(x_k)\rangle - \psi_2(\|x-x_k\|_2)$$
$$\langle x-x_k, f(x)\rangle f_j(x_k) - \frac{\psi_1'(\|x-x_k\|_2)}{\|x-x_k\|_2} (x-x_k)_j \langle f(x), f(x_k)\rangle - \psi_1(\|x-x_k\|_2)$$
$$\left. \left(\sum_{l=1}^{d} \frac{\partial f_l(x)}{\partial x_j} f_l(x_k) \right) \right)$$
$$= \sum_{k=1}^{N} \beta_k \left(\psi_3(\|x-x_k\|_2) (x-x_k)_j \langle x-x_k, f(x)\rangle \langle x_k-x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) f_j(x) \right.$$
$$\langle x_k-x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) \left(\sum_{l=1}^{d} (x-x_k)_l \frac{\partial f_l(x)}{\partial x_j} \right) \langle x_k-x, f(x_k)\rangle - \psi_2(\|x-x_k\|_2)$$
$$\langle x-x_k, f(x)\rangle f_j(x_k) - \psi_2(\|x-x_k\|_2) (x-x_k)_j \langle f(x), f(x_k)\rangle - \psi_1(\|x-x_k\|_2)$$
$$\left. \left(\sum_{l=1}^{d} \frac{\partial f_l(x)}{\partial x_j} f_l(x_k) \right) \right). \tag{5.7}$$

Then we take the absolute value on both sides

$$\left| \frac{\partial v'(x)}{\partial x_j} \right| \leq \sum_{k=1}^{N} |\beta_k| \left(|\psi_3(\|x-x_k\|_2)| \, |(x-x_k)_j| \, \|x-x_k\|_2^2 \, \|f(x)\|_2 \, \|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)| \right.$$
$$|f_j(x)| \, \|f(x_k)\|_2 \, \|x-x_k\|_2 + |\psi_2(\|x-x_k\|_2)| \left(\sum_{l=1}^{d} |(x-x_k)_l| \, |\frac{\partial f_l(x)}{\partial x_j}| \right) \|x-x_k\|_2$$
$$\|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)| \, \|x-x_k\|_2 \, \|f(x)\|_2 \, |f_j(x_k)| + |\psi_2(\|x-x_k\|_2)| \, |(x-x_k)_j|$$
$$\left. \|f(x)\|_2 \, \|f(x_k)\|_2 + |\psi_1(\|x-x_k\|_2)| \left(\sum_{l=1}^{d} |\frac{\partial f_l(x)}{\partial x_j}| \, |f_l(x_k)| \right) \right).$$

This gives

$$\left| \frac{\partial v'(x)}{\partial x_j} \right| \leq \boldsymbol{\beta} \Big(\Psi_3 r^3 F^2 + \Psi_2 r F^2 + \Psi_2 r^2 F D_1 + \Psi_2 r F^2 + \Psi_2 r F^2 + \Psi_1 F D_1 \Big)$$
$$= \boldsymbol{\beta} \Big(\Psi_3 r^3 F^2 + 3\Psi_2 r F^2 + \Psi_2 r^2 F D_1 + \Psi_1 F D_1 \Big).$$

□

Estimate (5.5) gives an upper bound on the first derivative of v' depending on ψ_i, $i \leq 3$, f, the first derivative of f and β_k, $k = 1, \ldots, N$, where β is the solution of the linear system

(2.7). Note that we only need a computer to calculate β, otherwise all theses quantities can be computed by hand.

Remark 5.2.
- *The product functions $\Psi_{i,k}$ are the maximum of the product of a decreasing function $\psi_i(r)$ and an increasing power function r^k, thus they are bounded and have a maximum.*
- *These product functions will differ according to the radial basis function used to construct the Lyapunov function v, and thus so will the corresponding formulas of the first and second derivatives of the orbital derivative.*

Here we will show how to calculate the product functions $\Psi_{i,k}$ with the Wendland basis functions and the Gaussian radial basis function. Similarly, this process can be applied to any radial basis function.

The product functions w.r.t the Wendland functions:

We fix the Wendland function $\psi_{6,4}$, and calculate the derivatives ψ_i, $i = 1, \ldots, 4$:

$$\begin{aligned}
\psi_{6,4}(r) \coloneqq \psi_0(r) &= (1-cr)_+^{10}[25 + 250cr + 1050c^2r^2 + 2250c^3r^3 + 2145c^4r^4].\\
\psi_1(r) &= -130c^2(1-cr)_+^{9}[5 + 45cr + 159c^2r^2 + 231c^3r^3].\\
\psi_2(r) &= 17160c^4(1-cr)_+^{8}[1 + 8cr + 21c^2r^2].\\
\psi_3(r) &= -514800c^6(1-cr)_+^{7}(1 + 7cr).\\
\psi_4(r) &= 28828800c^8(1-cr)_+^{6}.
\end{aligned}$$

then substitute back in (5.5)

$$\begin{aligned}
\left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(\Psi_{3,3}\,F^2 + \Psi_{2,2}\,F\,D_1 + 3\,\Psi_{2,1}\,F^2 + \Psi_{1,0}\,F\,D_1\Big) \\
&= \boldsymbol{\beta}\Big(\sup_{r\in[0,\infty)}|\psi_3(r)|\cdot r^3\,F^2 + \sup_{r\in[0,\infty)}|\psi_2(r)|\cdot r^2\,F\,D_1 + 3\sup_{r\in[0,\infty)}|\psi_2(r)|\cdot r\,F^2 \\
&+ \sup_{r\in[0,\infty)}|\psi_1(r)|\,F\,D_1\Big) \\
&= \boldsymbol{\beta}\Big(514800\,c^6\underbrace{\sup_{r\in[0,\infty)}(1-cr)_+^7(1+7cr)\,r^3}_{\Phi_2}\,F^2 \\
&+ 17160\,c^4\underbrace{\sup_{r\in[0,\infty)}(1-cr)_+^8(1+8cr+21c^2r^2)\,r^2}_{\Phi_4}\,F\,D_1 \\
&+ 51480\,c^4\underbrace{\sup_{r\in[0,\infty)}(1-cr)_+^8(1+8cr+21c^2r^2)\,r}_{\Phi_5}\,F^2 \\
&+ 130\,c^2\underbrace{\sup_{r\in[0,\infty)}(1-cr)_+^9(5+45cr+159c^2r^2+231c^3r^3)}_{\Phi_7}\,FD_1\Big). \qquad (5.8)
\end{aligned}$$

We find the maximum of the product functions $\Psi_{i,k}$, (for the full calculations see A.1).

$$\begin{aligned}
\Psi_{3,3} &= 514800\ (\Phi_2) = 514800\ (0.007253\ \frac{1}{c^3}) = 3733.84\ \frac{1}{c^3}, \\
\Psi_{2,2} &= 17160\ (\Phi_4) = 17160\ (0.027667\ \frac{1}{c^2}) = 474.77\ \frac{1}{c^2}, \\
\Psi_{2,1} &= 51480\ (\Phi_5) = 51480\ (0.115492\ \frac{1}{c}) = 5945.53\ \frac{1}{c}, \\
\Psi_{1,0} &= 130\ (\Phi_7) = 130\ (5) = 650.
\end{aligned}$$

Finally, we substitute the values of the product functions into (5.8), which will produce a simplified version of (5.8) which looks like

$$\begin{aligned}
\left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(3733.84\,c^3F^2 + 474.77\,c^2FD_1 + 5945.53\,c^3F^2 + 650\,c^2FD_1\Big) \\
&= \boldsymbol{\beta}\Big(9679.37\,c^3F^2 + 1124.77\,c^2FD_1\Big). \qquad (5.9)
\end{aligned}$$

The product functions w.r.t the Gaussian functions:

First, we state the formulas of the Gaussian RBF with parameter $\varepsilon > 0$ and the

derivatives $\psi_i,\ i = 1, \ldots, 4$:

$$\begin{aligned}
\psi_0(r) &= e^{-\varepsilon^2\, r^2}. \\
\psi_1(r) &= -2\,\varepsilon^2\, e^{-\varepsilon^2\, r^2}. \\
\psi_2(r) &= 4\,\varepsilon^4\, e^{-\varepsilon^2\, r^2}. \\
\psi_3(r) &= -8\,\varepsilon^6\, e^{-\varepsilon^2\, r^2} \\
\psi_4(r) &= 16\,\varepsilon^8\, e^{-\varepsilon^2\, r^2}.
\end{aligned}$$

then substitute in (5.5)

$$\begin{aligned}
\left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(\Psi_{3,3}\, F^2 + \Psi_{2,2}\, F\, D_1 + 3\,\Psi_{2,1}\, F^2 + \Psi_{1,0}\, F\, D_1\Big) \\
&= \boldsymbol{\beta}\Big(\sup_{r\in[0,\infty)} |\psi_3(r)| \cdot r^3\, F^2 + \sup_{r\in[0,\infty)} |\psi_2(r)| \cdot r^2\, F\, D_1 + 3 \sup_{r\in[0,\infty)} |\psi_2(r)| \cdot r\, F^2 \\
&+ \sup_{r\in[0,\infty)} |\psi_1(r)|\, F\, D_1\Big) \\
&= \boldsymbol{\beta}\Big(8\,\varepsilon^6 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2\, r^2}\, r^3}_{\Phi_2} F^2 + 4\,\varepsilon^4 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2\, r^2}\, r^2}_{\Phi_3}\ FD_1 + 12\,\varepsilon^4 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2\, r^2}\, r}_{\Phi_4}\ F^2 \\
&+ 2\,\varepsilon^2 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2\, r^2}}_{\Phi_5}\ FD_1\Big). \qquad (5.10)
\end{aligned}$$

We find the maximum of the product functions $\Psi_{i,k}$, (for the full calculations see A.2).

$$\begin{aligned}
\Psi_{3,3} &= 8\ (\Phi_2) = 8\ (0.4099\ \frac{1}{\varepsilon^3}) = 3.256\ \frac{1}{\varepsilon^3}, \\
\Psi_{2,2} &= 4\ (\Phi_3) = 4\ (0.3679\ \frac{1}{\varepsilon^2}) = 1.472\ \frac{1}{\varepsilon^2}, \\
\Psi_{2,1} &= 12\ (\Phi_4) = 12\ (0.4289\ \frac{1}{\varepsilon}) = 5.148\ \frac{1}{\varepsilon}, \\
\Psi_{1,0} &= 2\ (\Phi_5) = 2.
\end{aligned}$$

Finally, we substitute the values of the product functions into (5.10), which will produce a simplified version of (5.10) looks like

$$\begin{aligned}
\left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(3.256\,\varepsilon^3 F^2 + 1.472\,\varepsilon^2 FD_1 + 5.148\,\varepsilon^3 F^2 + 2\,\varepsilon^2 FD_1\Big) \\
&= \boldsymbol{\beta}\Big(8.404\,\varepsilon^3 F^2 + 3.472\,\varepsilon^2 FD_1\Big). \qquad (5.11)
\end{aligned}$$

Theorem 5.3 (The second derivative of the orbital derivative)**.** *Let $v \in C^3(\mathbb{R}^d, \mathbb{R})$ be an RBF approximant as in (2.8) with a radial basis function $\psi := \psi_0(r) \in C^4$. Let K be a compact set and $\{x_1, \dots, x_N\} \subseteq K$ be a set of N pairwise distinct points which are no equilibria. Then the second derivative of the orbital derivative v' can be bounded by*

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq \boldsymbol{\beta}\Big(\Psi_{4,4}\, F^2 + 6\Psi_{3,2}\, F^2 + 2\Psi_{3,3}\, F D_1 + 6\Psi_{2,1}\, F D_1 + 3\Psi_{2,0}\, F^2 + \Psi_{2,2}\, F D_2 + \Psi_{1,0} F D_2\Big), \tag{5.12}$$

$\forall\, x \in K$, where:

- $\Psi_{i,k} := \sup\limits_{r \in [0,\infty)} |\psi_i(r)| \cdot r^k$, *$i = 1, 2, 3, 4$, and $k \in \mathbb{N}_0^+$.*
- *The functions $\psi_i(r)$ are defined as in (5.6).*
- $F := \max\limits_{x \in K} \|f(x)\|_2$.
- $D_1 := \max\limits_{x \in K} \max\limits_{j \in 1,\dots,d} \|\frac{\partial f(x)}{\partial x_j}\|_2$.
- $D_2 := \max\limits_{x \in K} \max\limits_{i,j \in 1,\dots,d} \|\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\|_2$
- $\boldsymbol{\beta} := \sum_{k=1}^{N} |\beta_k|$.

Proof. We differentiate equation (5.7) stated in the proof of Theorem 5.2 as follows

$$\begin{aligned}
\frac{\partial^2 v'(x)}{\partial x_i \partial x_j} = \sum_{k=1}^{N} \beta_k \Bigg(& \frac{\psi_3'(\|x-x_k\|_2)}{\|x-x_k\|_2}(x-x_k)_i (x-x_k)_j \langle x-x_k, f(x)\rangle \langle x_k - x, f(x_k)\rangle \\
& + \psi_3(\|x-x_k\|_2)\, \delta_{ij}\, \langle x-x_k, f(x)\rangle \langle x_k - x, f(x_k)\rangle + \psi_3(\|x-x_k\|_2)(x-x_k)_j\, f_i(x) \\
& \langle x_k - x, f(x_k)\rangle + \psi_3(\|x-x_k\|_2)\, (x-x_k)_j \Bigg(\sum_{l=1}^{d} (x-x_k)_l\, \frac{\partial f_l(x)}{\partial x_i} \Bigg) \langle x_k - x, f(x_k)\rangle \\
& - \psi_3(\|x-x_k\|_2)\, (x-x_k)_j\, \langle x-x_k, f(x)\rangle f_i(x_k) + \frac{\psi_2'(\|x-x_k\|_2)}{\|x-x_k\|_2}(x-x_k)_i\, f_j(x) \\
& \langle x_k - x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2)\, \frac{\partial f_j(x)}{\partial x_i}\, \langle x_k - x, f(x_k)\rangle - \psi_2(\|x-x_k\|_2) f_j(x) f_i(x_k) \\
& + \frac{\psi_2'(\|x-x_k\|_2)}{\|x-x_k\|_2}(x-x_k)_i \Bigg(\sum_{l=1}^{d} (x-x_k)_l\, \frac{\partial f_l(x)}{\partial x_j} \Bigg) \langle x_k - x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) \\
& \frac{\partial f_i(x)}{\partial x_j}\, \langle x_k - x, f(x_k)\rangle + \psi_2(\|x-x_k\|_2) \Bigg(\sum_{l=1}^{d} (x-x_k)_l\, \frac{\partial^2 f_l(x)}{\partial x_i \partial x_j} \Bigg) \langle x_k - x, f(x_k)\rangle \\
& - \psi_2(\|x-x_k\|_2) \Bigg(\sum_{l=1}^{d} (x-x_k)_l\, \frac{\partial f_l(x)}{\partial x_j} \Bigg)\, f_i(x_k)\, - \frac{\psi_2'(\|x-x_k\|_2)}{\|x-x_k\|_2}(x-x_k)_i\, f_j(x_k) \\
& \langle x-x_k, f(x)\rangle - \psi_2(\|x-x_k\|_2)\, f_j(x_k)\, f_i(x) - \psi_2(\|x-x_k\|_2) f_j(x_k) \Bigg(\sum_{l=1}^{d} (x-x_k)_l\, \frac{\partial f_l(x)}{\partial x_i} \Bigg) \\
& - \frac{\psi_2'(\|x-x_k\|_2)}{\|x-x_k\|_2}\, (x-x_k)_i\, (x-x_k)_j\, \langle f(x), f(x_k)\rangle - \psi_2(\|x-x_k\|_2)\, \delta_{ij}\, \langle f(x), f(x_k)\rangle \\
& - \psi_2(\|x-x_k\|_2)\, (x-x_k)_j \Bigg(\sum_{l=1}^{d} f_l(x_k)\, \frac{\partial f_l(x)}{\partial x_i} \Bigg) - \frac{\psi_1'(\|x-x_k\|_2)}{\|x-x_k\|_2}\, (x-x_k)_i \\
& \Bigg(\sum_{l=1}^{d} f_l(x_k)\, \frac{\partial f_l(x)}{\partial x_j} \Bigg) - \psi_1(\|x-x_k\|_2) \Bigg(\sum_{l=1}^{d} f_l(x_k)\, \frac{\partial^2 f_l(x)}{\partial x_i \partial x_j} \Bigg) \Bigg).
\end{aligned}$$

Taking the absolute value on both sides gives

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq \sum_{k=1}^{N} |\beta_k| \Bigg(|\psi_4(\|x-x_k\|_2)|\ |(x-x_k)_i|\ |(x-x_k)_j|\ \|x-x_k\|_2\ \|f(x)\|_2\ \|x_k-x\|_2$$

$$\|f(x_k)\|_2 + |\psi_3(\|x-x_k\|_2)|\ \|x-x_k\|_2\ \|f(x)\|_2\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_3(\|x-x_k\|_2)|$$

$$|(x-x_k)_j|\ |f_i(x)|\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_j|$$

$$\Bigg(\sum_{l=1}^{d} |(x-x_k)_l|\ |\frac{\partial f_l(x)}{\partial x_i}|\Bigg)\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_j|\ \|x-x_k\|_2$$

$$\|f(x)\|_2\ |f_i(x_k)| + |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_i|\ |f_j(x)|\ \|x_k-x\|_2\ \|f(x_k)\|_2$$

$$+ |\psi_2(\|x-x_k\|_2)|\ \Big|\frac{\partial f_j(x)}{\partial x_i}\Big|\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)|\ |f_j(x)|\ |f_i(x_k)|$$

$$+ |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_i|\ \Bigg(\sum_{l=1}^{d} |(x-x_k)_l|\ |\frac{\partial f_l(x)}{\partial x_j}|\Bigg)\ \|x_k-x\|_2\ \|f(x_k)\|_2$$

$$+ |\psi_2(\|x-x_k\|_2)|\ \Big|\frac{\partial f_i(x)}{\partial x_j}\Big|\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)|$$

$$\Bigg(\sum_{l=1}^{d} |(x-x_k)_l|\ \Big|\frac{\partial^2 f_l(x)}{\partial x_i \partial x_j}\Big|\Bigg)\ \|x_k-x\|_2\ \|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)|$$

$$\Bigg(\sum_{l=1}^{d} |(x-x_k)_l|\ |\frac{\partial f_l(x)}{\partial x_j}|\Bigg)\ |f_i(x_k)| + |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_i|\ |f_j(x_k)|\ \|x-x_k\|_2$$

$$\|f(x)\|_2 + |\psi_2(\|x-x_k\|_2)|\ |f_j(x_k)|\ |f_i(x)| + |\psi_2(\|x-x_k\|_2)|\ |f_j(x_k)|$$

$$\Bigg(\sum_{l=1}^{d} |(x-x_k)_l|\ \Big|\frac{\partial f_l(x)}{\partial x_i}\Big|\Bigg) + |\psi_3(\|x-x_k\|_2)|\ |(x-x_k)_i|\ |(x-x_k)_j|\ \|f(x)\|_2\ \|f(x_k)\|_2$$

$$+ |\psi_2(\|x-x_k\|_2)|\ \|f(x)\|_2\ \|f(x_k)\|_2 + |\psi_2(\|x-x_k\|_2)|\ |(x-x_k)_j|\ \Bigg(\sum_{l=1}^{d} |f_l(x_k)|\ \Big|\frac{\partial f_l(x)}{\partial x_i}\Big|\Bigg)$$

$$+ |\psi_2(\|x-x_k\|_2)|\ |(x-x_k)_i|\ \Bigg(\sum_{l=1}^{d} |f_l(x_k)|\ \Big|\frac{\partial f_l(x)}{\partial x_j}\Big|\Bigg) + |\psi_1(\|x-x_k\|_2)|$$

$$\Bigg(\sum_{l=1}^{d} |f_l(x_k)|\ \Big|\frac{\partial^2 f_l(x)}{\partial x_i \partial x_j}\Big|\Bigg)\Bigg).$$

Simplifying gives

$$\begin{aligned}\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq \boldsymbol{\beta}\Big(|\psi_4(\|x-x_k\|_2)|\ r^4F^2 + 6\ |\psi_3(\|x-x_k\|_2)|\ r^2F^2 + 2\ |\psi_3(\|x-x_k\|_2)|\ r^3FD_1 \\ &+ 6\ |\psi_2(\|x-x_k\|_2)|\ r\ F\ D_1 + 3\ |\psi_2(\|x-x_k\|_2)|F^2 + |\psi_2(\|x-x_k\|_2)|\ r^2FD_2 \\ &+ |\psi_1(\|x-x_k\|_2)|\ FD_2\Big) \\ &\leq \boldsymbol{\beta}\Big(\Psi_{4,4}\ F^2 + 6\ \Psi_{3,2}\ F^2 + 2\ \Psi_{3,3}\ FD_1 + 6\ \Psi_{2,1}\ FD_1 + 3\ \Psi_{2,0}\ F^2 + \Psi_{2,2}\ FD_2 \\ &+ \Psi_{1,0}\ FD_2\Big). \end{aligned} \tag{5.13}$$

□

Estimate (5.13) gives an upper bound of the second derivative of v' depending on ψ_i, $i = 1, 2, 3, 4$, f, the first and the second derivatives of f and β_k, $k = 1, \ldots, N$.
Now, we follow the same strategy of the previous estimate to get the simplified version of this estimate for the Wendland and the Gaussian basis functions.

The Wendland functions:

First, we substitute of the derivatives $\psi_i,\ i = 1, 2, 3, 4$ of the Wendland function $\psi_{6,4}$ in (5.13)

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq \boldsymbol{\beta}\Big(28828800\, c^8 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^6\, r^4}_{\Phi_1}\; F^2 + 3088800\, c^6 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^7(1+7cr)\, r^2}_{\Phi_3}\; F^2$$
$$+ 1029600\, c^6 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^7(1+7cr)\, r^3}_{\Phi_2}\; F\, D_1$$
$$+ 102960\, c^4 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^8(1+8cr+21c^2r^2)\, r}_{\Phi_5}\; F\, D_1$$
$$+ 51480\, c^4 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^8(1+8cr+21c^2r^2)}_{\Phi_6}\; F^2$$
$$+ 17160\, c^4 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^8(1+8cr+21c^2r^2)\, r^2}_{\Phi_4}\; F\, D_2$$
$$+ 130\, c^2 \underbrace{\sup_{r\in[0,\infty)} (1-cr)_+^9(5+45cr+159c^2r^2+231c^3r^3)}_{\Phi_7}\; F\, D_2\Big). \tag{5.14}$$

Then, we calculate the values of the product functions

$$\begin{aligned}
\Psi_{4,4} &= 28828800\ (\Phi_1) = 28828800\ (0.001194\frac{1}{c^4}) = 34421.5872\ \frac{1}{c^4}.\\
\Psi_{3,3} &= 1029600\ (\Phi_2) = 1029600\ (0.007253\frac{1}{c^3}) = 7467.6000\ \frac{1}{c^3}.\\
\Psi_{3,2} &= 3088800\ (\Phi_3) = 3088800\ (0.0233\frac{1}{c^2}) = 71969.04\ \frac{1}{c^2}.\\
\Psi_{2,2} &= 17160\ (\Phi_4) = 17160\ (0.027667\frac{1}{c^2}) = 474.796608\ \frac{1}{c^2}.\\
\Psi_{2,1} &= 102960\ (\Phi_5) = 102960\ (0.115492\frac{1}{c}) = 11899.73585\ \frac{1}{c}.\\
\Psi_{2,0} &= 51480\ (\Phi_6) = 51480.\\
\Psi_{1,0} &= 130\ (\Phi_7) = 130(5) = 650.
\end{aligned}$$

Finally, the simplified version of (5.14) will be

$$\begin{aligned}
\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq \boldsymbol{\beta}\Big(34421.5872\, c^4F^2 + 71969.04\, c^4F^2 + 7467.6888\, c^3FD_1 + 11891.056\, c^3D_1F\\
&\quad + 51480\, c^4F^2 + 474.796608\, c^2D_2F + 650\, c^2FD_2\Big)\\
&= \boldsymbol{\beta}\big(157870.623\, c^4F^2 + 19358.745\, c^3FD_1 + 1124.797\, c^2FD_2\big).
\end{aligned} \tag{5.15}$$

The Gaussian function:

Substitute of the derivatives $\psi_i,\ i = 1, 2, 3, 4$ of the Gaussian RBF function in (5.13)

$$\begin{aligned}\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq \boldsymbol{\beta}\Big(16\,\varepsilon^8 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2} r^4}_{\Phi_1} F^2 + 48\,\varepsilon^6 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2} r^2}_{\Phi_3} F^2 + 16\,\varepsilon^6 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2} r^3}_{\Phi_2} FD_1 \\ + 24\,\varepsilon^4 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2} r}_{\Phi_4} FD_1 + 12\,\varepsilon^4 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2}}_{\Phi_5} F^2 + 4\,\varepsilon^4 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2} r^2}_{\Phi_3} FD_2 \\ + 2\,\varepsilon^2 \underbrace{\sup_{r\in[0,\infty)} e^{-\varepsilon^2 r^2}}_{\Phi_5} FD_2\Big). \qquad (5.16)\end{aligned}$$

Then, we calculate the values of the product functions

$$\begin{aligned}\Psi_{4,4} &= 16\,(\Phi_1) = 16\,(0.5412\frac{1}{\varepsilon^4}) = 8.6592\,\frac{1}{\varepsilon^4}.\\ \Psi_{3,3} &= 16\,(\Phi_2) = 16\,(0.4099\frac{1}{\varepsilon^3}) = 6.512\,\frac{1}{\varepsilon^3}.\\ \Psi_{3,2} &= 48\,(\Phi_3) = 48\,(0.3679\frac{1}{\varepsilon^2}) = 17.664\,\frac{1}{\varepsilon^2}.\\ \Psi_{2,2} &= 4\,(\Phi_3) = 4\,(0.3679\frac{1}{\varepsilon^2}) = 1.472\,\frac{1}{\varepsilon^2}.\\ \Psi_{2,1} &= 24\,(\Phi_4) = 24\,(0.4289\frac{1}{\varepsilon}) = 10.296\,\frac{1}{\varepsilon}.\\ \Psi_{2,0} &= 12\,(\Phi_5) = 12.\\ \Psi_{1,0} &= 2\,(\Phi_5) = 2.\end{aligned}$$

Finally, the simplified version of (5.16) will be

$$\begin{aligned}\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq \boldsymbol{\beta}\Big(8.6592\,\varepsilon^4 F^2 + 17.664\,\varepsilon^4 F^2 + 6.512\,\varepsilon^3 FD_1 + 10.296\,\varepsilon^3 D_1 F \\ &\quad + 12\,\varepsilon^4 F^2 + 1.472\,\varepsilon^2 D_2 F + 2\,\varepsilon^2 FD_2\Big) \\ &= \boldsymbol{\beta}\big(38.3232\,\varepsilon^4 F^2 + 16.808\,\varepsilon^3 FD_1 + 3.472\,\varepsilon^2 FD_2\big). \qquad (5.17)\end{aligned}$$

5.2 Improvements of the estimates

In the previous section we have derived the general formulation of the verification estimates (5.2) and (5.4), where we have used the Euclidean norm to estimate the terms

$\underbrace{\max_{z\in co(K)} \|\nabla v'(z)\|_p \, h_{\|.\|_q}}_{B_1}$ and $\underbrace{\max_{z\in\mathcal{T}} \|H(z)\|_p \, h^2_{\|.\|_q}}_{B_2}$, with $p = q = 2$. As we mentioned in Remark 5.1, we have obtained a bound on the $L_\infty \backslash L_{\max}$ norm of the quantities $\nabla v'(z)$ and $H(z)$, rather than on the L_2-norm. Therefore, the first idea towards improving the estimates is by changing the norms p and q. This will be done by using different combinations of p-norms, satisfying the condition $\frac{1}{p} + \frac{1}{q} = 1$, where $1 \leq p, q \leq \infty$.

On the other hand, since the value of $h_{\|.\|_q}$ in both terms B_1 and B_2 varies according to the structure of the grid points as well as the norm used to calculate it, thus the other direction to be investigated is the different distributions of grid points in the checking grid Y_{od}. In particular, we consider two structures of grid points, namely, the square configuration and the body centred square configuration. Then we cover each structure with p-norms balls, to find the value of $h_{\|.\|_q}$ under each norm $q = 1, 2, \infty$. Obviously, the terms B_1 and B_2 will take different forms based on the values of $h_{\|.\|_q}$ under different norms. As a result, we choose the best structure of points in terms of the value of $h_{\|.\|_q}$ which minimises each term and altogether requires least points.

In this section, we will study each factor in details and see how they affect the final formulation of the verification estimates.

5.2.1 Factor One: Distance Functions (Norms)

Most of our studies in this section depend on some fundamental material related to vector norms. Therefore, we start with a small introduction to norms including their definition, equivalence of norms and the concept of the unit-ball under different norms.

The L_p Vector Norms:

The vector norms allows to measure the length of a vector x in $\mathbb{R}^d$, this possessing the following characteristic properties:

Definition 5.2. *A norm is a scalar-valued function $\|.\| : \mathbb{R}^d \to \mathbb{R}$, such that*

1. $\|x\| \geq 0$, $\forall x \in \mathbb{R}^d$, *and* $\|x\| = 0$ *if and only if* $x = 0$. *(positivity)*
2. $\|cx\| = |c| \, \|x\|$, *for any* $c \in \mathbb{R}$, $x \in \mathbb{R}^d$. *(homogeneity)*
3. $\|x + y\| \leq \|x\| + \|y\|$, *for any* x, $y \in \mathbb{R}^d$. *(triangle inequality)*

The most popular norms on $\mathbb{R}^d$ are the so called L_p-norms, which are defined as:

$$\|x\|_p = \left(\sum_{i=1}^{d} |x_i|^p \right)^{\frac{1}{p}} \quad \text{for } x \in \mathbb{R}^d \text{ and } 1 \leq p \leq \infty$$

and the three commonly used L_p-norms in applications are:

1. The L_1-norm ($p = 1$, known as the Taxicab norm):

$$\|x\|_1 = \sum_{i=1}^{d} |x_i|.$$

2. The L_2-norm ($p = 2$, known as the Euclidean norm):

$$\|x\|_2 = \left(\sum_{i=1}^{d} |x_i|^2 \right)^{\frac{1}{2}}.$$

3. The L_∞-norm ($p = \infty$, known as the Maximum Uniform norm):

$$\|x\|_\infty = \max_{1 \leq i \leq d} \left(|x_i| \right).$$

Note: Different norms are just different ways of measuring distances in $\mathbb{R}^d$.

Equivalence of norms:

A useful fact about norms is that they are all equivalent on $\mathbb{R}^d$. This means, given any pairs of norms $\|.\|_{\alpha_1}$ and $\|.\|_{\alpha_2}$, there exist positive real constants $0 < C_1 \leq C_2$ such that

$$C_1 \|x\|_{\alpha_1} \leq \|x\|_{\alpha_2} \leq C_2 \|x\|_{\alpha_1}, \ \forall \ x \in \mathbb{R}^d.$$

In particular, the following inequalities hold for all $x \in \mathbb{R}^d$:

$$\begin{aligned} \|x\|_2 \leq \|x\|_1 &\leq \sqrt{d} \, \|x\|_2 \\ \|x\|_\infty \leq \|x\|_2 &\leq \sqrt{d} \, \|x\|_\infty \\ \|x\|_\infty \leq \|x\|_1 &\leq d \, \|x\|_\infty \end{aligned} \tag{5.18}$$

The unit-ball in different norms:

The unit-ball is the set of all vectors that have length ≤ 1, and it is defined as

$$\mathcal{B}_p = \{x \in \mathbb{R}^d \mid \|x\|_p \leq 1\}.$$

The unit-ball is a convex, closed, bounded, and centrally symmetric set that has different shapes in different norms. In Table 5.1 we illustrate the different shapes of the unit-ball

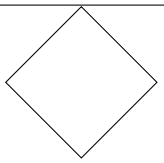
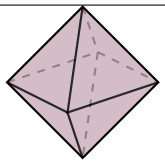
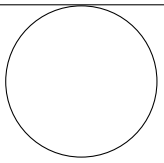
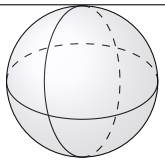
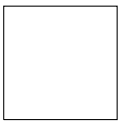
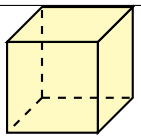
Unit-ball	d=2	d=3
$\mathcal{B}_1$ (L_1-norm)		
$\mathcal{B}_2$ (L_2-norm)		
$\mathcal{B}_\infty$ (L_∞-norm)		

Table 5.1: The different shapes of the unit-ball under the 1, 2, and ∞ norms in 2 and 3 dimensions.

$\mathcal{B}_p$ in $d = 2$ and $d = 3$ for the L_1, L_2, and L_∞ norms.

The L_P Matrix Norms:

Measuring the "size" of a matrix is also possible by using matrix norms.

Definition 5.3 (Matrix norm)**.** *The norm of a square matrix $A \in \mathbb{R}^{d\times d}$ is a function $\|.\| : \mathbb{R}^{d\times d} \to \mathbb{R}$, that satisfies the following axioms:*

1. $\|A\| \geq 0, \forall\ A \in \mathbb{R}^{d\times d}$ *and* $\|A\| = 0$ *if and only if* $A = 0$. *(positivity)*
2. $\|cA\| = |c|\ \|A\|$, $\forall c \in \mathbb{R},\ A \in \mathbb{R}^{d\times d}$ *(homogeneity)*
3. $\|A + B\| \leq \|A\| + \|B\|$, *for all* $A, B \in \mathbb{R}^{d\times d}$. *(triangle inequality)*

The most popular matrix norms are the L_p norms that correspond to the L_p vector norms and defined as

$$\|A\|_p = \max_{\|x\|_p=1} \|Ax\|_p.$$

For $p = 1, 2, \infty$ we have

1. The L_1 matrix norm :

$$\|A\|_1 = \max_{1\leq j\leq d} \left(\sum_{i=1}^{d} \left| a_{ij} \right| \right).$$

2. The L_2 matrix norm :

$$\|A\|_2 = \left(\sum_{i=1}^{d} \sum_{j=1}^{d} (a_{ij})^2 \right)^{\frac{1}{2}}.$$

3. The L_∞ matrix norm :

$$\|A\|_\infty = \max_{1 \le i \le d} \left(\sum_{j=1}^{d} \left| a_{ij} \right| \right).$$

4. The $L_{\max}$ matrix norm :

$$\|A\|_{\max} = \max_{1 \le i,j \le d} |a_{ij}|.$$

5.2.2 Factor Two: Distribution of Grid points

The main and most important goal of this section is to find the fill distance $h_{\|.\|_q}$ of certain types of the checking grid Y_{od} under different norms, i.e., $q = 1, 2, \infty$. Let the points in Y_{od} be distributed in either a square or a body centred square structures, such that each point $y \in Y_{od}$ is the centre of a p-norm ball of radius R_p, which contains all the points $x \in K$ that are closer to this centre y than the other centres in Y_{od}.

This process will divide the set Y_{od} into symmetrical regions depending on the shape of the specified p-norm ball. For example, in $\mathbb{R}^2$, $\mathcal{B}_1$ is a diamond square, $\mathcal{B}_2$ is a circle and $\mathcal{B}_\infty$ is a square. While in $\mathbb{R}^3$, $\mathcal{B}_1$ is a octahedron, $\mathcal{B}_2$ is a sphere and $\mathcal{B}_\infty$ is a cube, etc, see Table 5.1.

Therefore, to determine the fill distance of each structure, we need to find the optimal value of R_p, such that the balls $\mathcal{B}_p$, centred at each point of Y_{od}, cover the whole set K with no gaps between them. For instance, if we want to find the fill distance $h_{\|.\|_1}$ of a 2-D square configuration under the L_1-norm, we cover the set with L_1-norm balls, (i.e., $\mathcal{B}_1$ are diamond squares), centred at each grid point and then determine the value of R_1 that allows the balls $\mathcal{B}_1$ to fully cover $\mathbb{R}^2$.

The rest of this section will give the mathematical definition of both the square and the body centred square configurations.

The Square Configuration:

The square configuration of grid points in $\mathbb{R}^d$ are a uniformly distributed points forming hypercubes and is mathematically generated by

$$SC = \{(x_1, x_2, \ldots, x_d) \in \mathbb{R}^d \mid \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = \begin{pmatrix} h_1 i_1 \\ h_1 i_2 \\ \vdots \\ h_1 i_d \end{pmatrix}, \ h_1 > 0, \text{ and } i_1, i_2, \ldots, i_d \in \mathbb{Z}\} \quad (5.19)$$

where h_1 is the distance between the grid points in every direction of $\mathbb{R}^d$. Figure 5.2 is an example of a square configuration in $\mathbb{R}^2$.

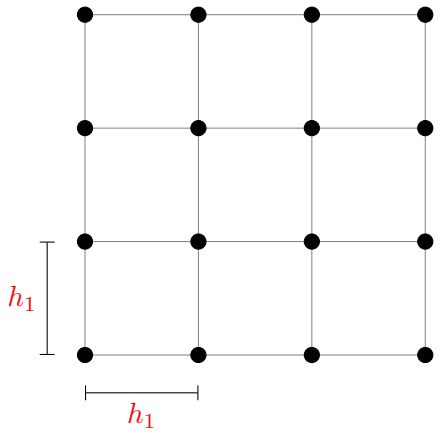


Figure 5.2: A square configuration of grid points in $\mathbb{R}^2$, h_1 is the distance between grid points in both directions.

The Body Centered Square Configuration:

The body centred square configuration of grid points in $\mathbb{R}^d$ is a square configuration (5.19) including the centres of the hypercubes. The centres are generated by

$$BCSC = SC \cup \Big\{SC + \frac{1}{2}\, h_1 \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \ h_1 > 0\Big\}. \quad (5.20)$$

An example of a body centred square configuration in $\mathbb{R}$ is illustrated in Figure 5.3.

Number of points in a box of square and body centred square configurations:

Let $H = [-l, l]^d \subset \mathbb{R}^d$, $l \in \mathbb{R}$, $l > 0$, be a hypercube in $\mathbb{R}^d$ such that $-l \leq x_1 \leq l$, $-l \leq x_2 \leq l, \ldots, -l \leq x_d \leq l$. Let the points in each direction of the space be equally separated with a distance h_1. Then,

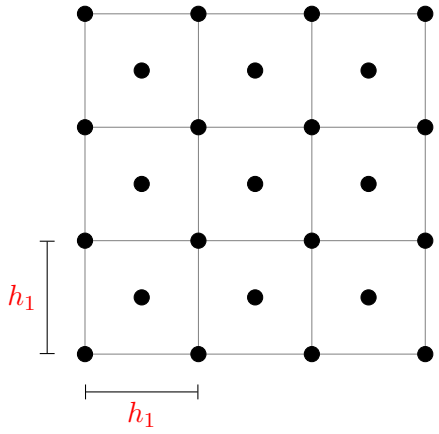


Figure 5.3: A body centred square configuration of grid points in $\mathbb{R}^2$.

1. The number of points in H arranged in a square order is given by

$$M(l) = \left\lfloor \frac{2l}{h_1} + 1 \right\rfloor^d, \tag{5.21}$$

2. The number of points in H arranged in a body centred square order is given by

$$M(l) = \left\lfloor \frac{2l}{h_1} + 1 \right\rfloor^d + \left\lfloor \frac{2l}{h_1} \right\rfloor^d. \tag{5.22}$$

In the following sections we will give a detailed study of the effects of the norms and grid points distributions on improving the outcome of the first and second verification estimates.

5.3 Final Formulation of the Estimates

This section investigates how these factors will affect the final formulation of the first and second verification estimates.

5.3.1 The first estimate (5.2)

Factor (1): Different norms

Consider the term $\underbrace{\max_{z\in co(K)} \|\nabla v'(z)\|_p \, h_{\|.\|_q}}_{B_1}$, where p and q are different L_p norms satisfying the condition $\frac{1}{p} + \frac{1}{q} = 1$ and $1 \leq p, q \leq \infty$. This condition along with the inequalities

Case ($\frac{1}{p}+\frac{1}{q}=1$)	Term(B_1)	Remark 5.1 + inequalities (5.18)
$p=q=2$	$\|\nabla v'(x)\|_2\ h_{\|.\|_2}$	$\|\nabla v'(x)\|_2 \le \sqrt{d}\ \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)$ $\Rightarrow\ v'(x) \le \max\limits_{y\in Y_{od}} v'(y) + \sqrt{d}\ \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)\ h_{\|.\|_2}$ (5.23)
$p=1, q=\infty$	$\|\nabla v'(x)\|_1\ h_{\|.\|_\infty}$	$\|\nabla v'(x)\|_1 \le d\ \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)$ $\Rightarrow\ v'(x) \le \max\limits_{y\in Y_{od}} v'(y) + d\ \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)\ h_{\|.\|_\infty}$ (5.24)
$p=\infty, q=1$	$\|\nabla v'(x)\|_\infty\ h_{\|.\|_1}$	$\|\nabla v'(x)\|_\infty \coloneqq \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)$ $\Rightarrow\ v'(x) \le \max\limits_{y\in Y_{od}} v'(y) + \left(\max\limits_{x\in K}\max\limits_{j=1\ldots d}\left\|\frac{\partial v'(x)}{\partial x_j}\right\|\right)\ h_{\|.\|_1}$ (5.25)

Table 5.2: The three different combinations of the L_1, L_2, and L_∞ norms and the corresponding formulas of both the term (B_1) and estimate (5.2) in each case.

(5.18) and Remark 5.1 enable us to identify three possible cases of the term (B_1) and accordingly we obtain three versions of estimate (5.2) as explained in Table 5.2.

The effects of using different combinations of norms on formulating the estimate, appear clearly in the three versions (5.23), (5.24), and (5.25), presented in Table 5.2. In principle, the third version (5.25) improves the original estimate by getting rid of the constants obtained in the other cases, which improves the outcome especially in higher dimensional cases. Therefore, we will only examine this case in the further studies. More precisely, we will take into account finding the fill distance $h_{\|.\|_1}$ under the L_1-norm only, for both the square and the body centred square structures of grid points.

Factor (2): Distributions of grid points

Let Y_{od} be a set of checking grid points, such that these points are structured as either a square or a body centred square forms. Then, to find the fill distance $h_{\|.\|_1}$ for both structures under the L_1-norm, we cover the whole set with L_1-norm balls of radius R_1 centred at each grid point. As we see in Figure 5.2 and Figure 5.3, both configurations consist of small symmetrical squares of size $h_1 \times h_1$ each, where h_1 is the distance between grid points in every direction of the space. Therefore, for simplicity, we will study only one square of each configurations, namely $\overline{S} = [0, h_1]^2$ and determine the fill distance $h_{\|.\|_1}$

with respect to it. The following theorems give the value of the fill distance $h_{\|.\|_1}$ for d-dimensional square and body centred square configurations under the L_1-norm. Moreover, it turns out that the values obtained of $h_{\|.\|_1}$ are actually in terms of h_1.

Theorem 5.4 (The fill distance for the d-dimensional square grid w.r.t the L_1-norm.)**.** *Let $\overline{S} = [0, h_1]^d \subset \mathbb{R}^d$ be a hypercube of an d-dimensional square grid, such that $S_v = \{a_i \in Y_{od} \mid a_i = (x_1, \ldots, x_d),\ i = 1, \ldots, 2^d\} \subset \overline{S}$ are the vertices of $\overline{S}$. Then the fill distance, the largest distance from any point $x \in \overline{S}$ to the nearest grid point $a_i \in S_v$, under the L_1-norm is given by*

$$h_{\|.\|_1} = \max_{x \in \overline{S}} \min_{\substack{a_i \in S_v \\ i=1,\ldots,2^d}} \|x - a_i\|_1 = \frac{d}{2}\, h_1,$$

where h_1 is the distance between grid points in all directions of $\mathbb{R}^d$.

Proof. Let $\overline{S} = [0, h_1]^d \subset \mathbb{R}^d$ be a hypercube of a square configuration with vertices $a_i = (x_1, \ldots, x_d) \in S_v,\ i = 1, \ldots, 2^d$, i.e., $a_1 = \underbrace{(0, 0, \ldots, 0)}_{\text{d-times}}, \ldots, a_{2^d} = \underbrace{(h_1, h_1, \ldots, h_1)}_{\text{d-times}}$.
Let R_1 be the radius of the L_1-norm ball centred at each vertex a_i of $\overline{S}$. Then our aim is to find the minimal R_1 that the balls fully cover $\overline{S}$.
Let $(x_1, x_2, \ldots, x_d) \in \overline{S}$, such that $x_1 \leq h_1$, $x_2 \leq h_1, \ldots, x_d \leq h_1$. Moreover, assume that

$$|h_1 - x_1| + |h_1 - x_2| + \ldots + |h_1 - x_d| > R_1. \tag{5.26}$$

Equation (5.26) means that, the point $(x_1, x_2, \ldots, x_d)$ is outside the L_1-norm ball centred at the grid point $a_{2^d} = \underbrace{(h_1, h_1, \ldots, h_1)}_{\text{d-times}}$. Then, solving (5.26) gives

$$\begin{aligned} d\, h_1 - (x_1 + x_2 + \ldots + x_d) &> R_1, \\ \Rightarrow x_1 + x_2 + \ldots + x_d &< d\, h_1 - R_1 \end{aligned} \tag{5.27}$$

On the other hand, (5.27) shows that the point $(x_1, x_2, \ldots, x_d)$ is inside the L_1-norm ball of the grid point $a_1 = \underbrace{(0, 0, \ldots, 0)}_{\text{d-times}}$ of radius $d\, h_1 - R_1$. Setting this radius equal to R_1 yields

$$\begin{aligned} d\, h_1 - R_1 = R_1 \Rightarrow d\, h_1 &= 2\, R_1, \\ \Rightarrow \frac{d}{2}\, h_1 &= R_1. \end{aligned}$$

On the other hand, there is a points, namely $\frac{h_1}{2}\ \underbrace{(1,1,\ldots,1)^T}_{d\text{-times}}$ with distance $\frac{d}{2}\ h_1$ to each grid point. Therefore, the minimum radius R_1 to cover $\overline{S}$ with L_1-norm balls is $R_1 = \frac{d}{2}\ h_1$. Moreover, it shows that if a point $(x_1, x_2, \ldots, x_d) \in \overline{S}$ is outside the L_1-norm ball of a grid point (a vertex) $a_i \in S_v$, it will definitely be inside the L_1-norm ball of one of the remaining vertices.

□

Remark 5.3. *For the square configuration we only need two L_1-norm balls to cover $\overline{S}$, see Figure 5.4 (b). This will in turn lead to an overlap between the L_1-norm balls of the vertices of $\overline{S}$. Therefore, to improve this situation we use the body centred square configuration.*

Theorem 5.5 (The fill distance for the d-dimensional body centred square grid in the L_1-norm)**.** *Let $\overline{S} = [0, h_1]^d \subset \mathbb{R}^d$ be a hypercube of an d- dimensional body centred square grid, such that $S_v = \{a_i \in Y_{od} \mid a_i = (x_1, \ldots, x_d),\ i = 1, \ldots, 2^d + 1\} \subset \overline{S}$ are the vertices and the centre of $\overline{S}$. Then the fill distance, the largest distance from any point $x \in \overline{S}$ to the nearest grid point $a_i \in S_v$, under the L_1-norm is given by*

$$h_{\|.\|_1} = \max_{x \in \overline{S}} \min_{\substack{a_i \in S_v \\ i=1,\ldots,2^d+1}} \|x - a_i\|_1 = \frac{d}{4}\ h_1,$$

Proof. Let $\overline{S} = [0, h_1]^d$ be a hypercube of a d-dimensional body centred square grid in $\mathbb{R}^d$ such that the grid points $\{a_i\}_{i=1}^{2^d+1} \in Y_{od} \subset \overline{S}$, represent the vertices and the center of $\overline{S}$. Let R_1 be the radius of the L_1-norm ball centred at each grid point $a_i, i = 1, \ldots, 2^d + 1$. Then, our goal is to find the minimum R_1 that covers the whole $\overline{S}$ with L_1-norm balls. Let $(x_1, x_2, \ldots, x_d) \in \overline{S}$, such that: $x_1 \leq \frac{h_1}{2}$, $x_2 \leq \frac{h_1}{2}, \ldots, x_d \leq \frac{h_1}{2}$, (the other corners are similar).

Assume

$$\left|\frac{h_1}{2} - x_1\right| + \left|\frac{h_1}{2} - x_2\right| + \ldots + \left|\frac{h_1}{2} - x_d\right| > R_1 \tag{5.28}$$

Meaning that, the point $(x_1, x_2, \ldots, x_d)$ is outside the L_1-norm ball of the centre grid point

$o = \underbrace{\left(\frac{h_1}{2}, \frac{h_1}{2}, \ldots, \frac{h_1}{2}\right)}_{\text{d-times}}$. Then, solving (5.28) gives

$$\begin{aligned} \frac{d}{2}h_1 - (x_1 + x_2 + \ldots + x_d) > R_1, \\ \Rightarrow \frac{d}{2}h_1 - R_1 > x_1 + x_2 + \ldots + x_d, \\ \Rightarrow x_1 + x_2 + \ldots + x_d < \frac{d}{2}\, h_1 - R_1, \end{aligned} \tag{5.29}$$

which also means that the point $(x_1, x_2, \ldots, x_d)$ lies inside the L_1-norm ball of the grid point $a_1 = \underbrace{(0, 0, \ldots, 0)}_{\text{d-times}}$ of radius $\frac{d}{2}\, h_1 - R_1$. Setting this radius equal to R_1 yields

$$\begin{aligned} \frac{d}{r}h_1 - R_1 = R_1, \\ \Rightarrow \frac{d}{2}h_1 = 2R_1, \\ \Rightarrow \frac{d}{4}h_1 = R_1. \end{aligned} \tag{5.30}$$

On the other hand, there is a point, namely $\frac{h_1}{4}\underbrace{(1, 1, \ldots, 1)^T}_{d\text{-times}}$ with distance $\frac{d}{4}\, h_1$ to the grid points o and a_1. Thus the minimal R_1 that fully covers $\overline{S}$ with L_1-norm balls is $\frac{d}{4}\, h_1$. Moreover, with this R_1 all the points $x \in \overline{S}$ are covered, i.e., if a points $(x_1, x_2, \ldots, x_d)$ is outside the L_1-norm ball of a grid point $a_i \in S_v$, it will be inside the L_1-norm ball of one of the other grid points in S_v. □

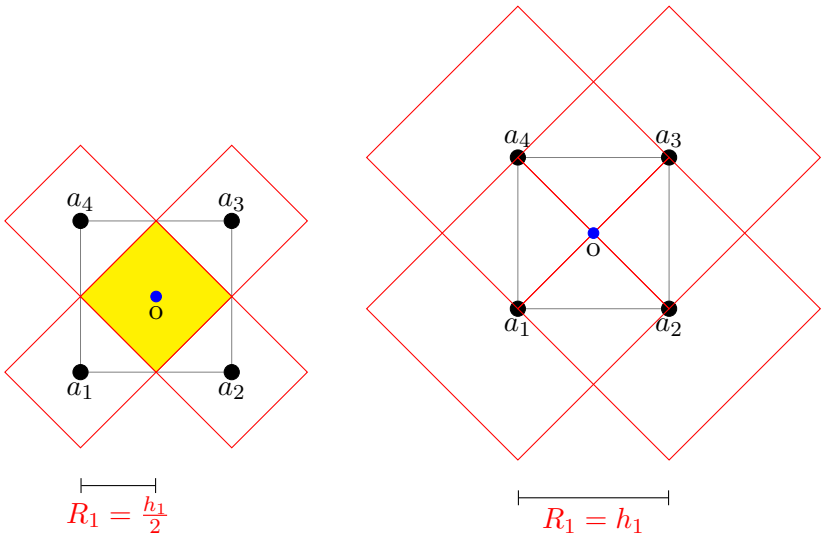
In Figure 5.4 and Figure 5.5, we explain how to cover a 2-D square $\overline{S} = [0, h_1]^2$ of a square and a body centred square configuration, with L_1-norm balls. Moreover, they show how the square is fully covered with the obtained value of R_1 in both cases.

The improved formula of the first estimate:

After investigating the effects of both factors in formulating the estimate (5.2), we will state the improved formula of the estimate in two versions based on the structure of the checking grid points Y_{od}.

Best configuration for the first estimate.

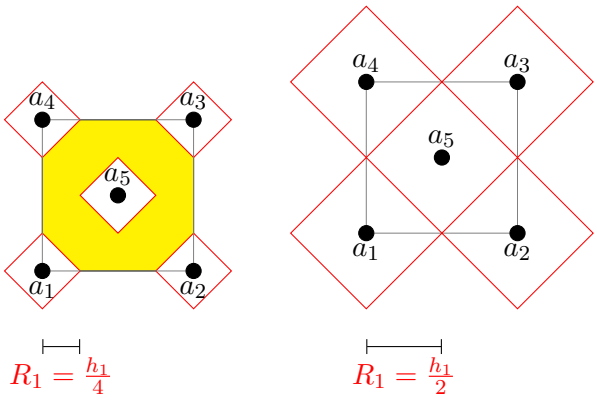
So far, the best grid configuration for the first estimate is the body centred square structure



(a) The L_1-norm balls with radius $R_1 = \frac{1}{2}h_1$.

(b) The L_1-norm balls with radius $R_1 = h_1$

Figure 5.4: (a) The uncovered area (yellow) when choosing L_1-norm balls of radius $R_1 < h_1$, (b) The square is completely covered with L_1-norm balls of radius $R_1 = h_1$, centred at the vertices.



(a) The L_1-norm balls with radius $R_1 = \frac{1}{4}h_1$.

(b) The L_1-norm balls with radius $R_1 = \frac{1}{2}h_1$

Figure 5.5: (a) The uncovered area (yellow) when choosing L_1-norm balls of radius $R_1 < \frac{1}{2}h_1$, (b) The square is completely covered with L_1-norm balls of radius $R_1 = \frac{1}{2}h_1$, centred at the vertices and the centre of the square.

Fill distance	d-D Square grid
$h_{\|.\|_1} = \frac{d}{2}h_1$	$v'(x) \leq \max\limits_{y \in Y_{od}} v'(y) + \frac{d}{2}(\max\limits_{x \in K} \max\limits_{j=1...d} \lvert\frac{\partial v'(x)}{\partial x_j}\rvert)\ h_1$ (5.31)

Table 5.3: The estimate formula for a d-D square grid, with the fill distance calculated w.r.t the L_1-norm.

Fill distance	d-D Body centred square grid
$h_{\|.\|_1} = \frac{d}{4}h_1$	$v'(x) \leq \max\limits_{y \in Y_{od}} v'(y) + \frac{d}{4}(\max\limits_{x \in K} \max\limits_{j=1...d} \lvert\frac{\partial v'(x)}{\partial x_j}\rvert)\ h_1$ (5.32)

Table 5.4: The estimate formula for a d-D body centred square grid, with the fill distance calculated w.r.t the L_1-norm.

as it provides the minimum value of $h_{\|.\|_1}$, see equation (5.32). Moreover, it turns out that this structure also requires less points than the square structure does, as we will see in the following theorem.

Theorem 5.6. *If the grid points in the checking grid Y_{od} are distributed in a box $K = [-1, 1]^d \subset \mathbb{R}^d$ of a body centred square order, then we need about $\frac{1}{2^{d-1}}$ times fewer points than if they are distributed in a square order for the same bound in the estimate.*

Proof. Recall the formula of the estimate for the square configuration in $\mathbb{R}^d$

$$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \frac{d}{2} \left(\max_{x \in K} \max_{j=1...d} \left| \frac{\partial v'(x)}{\partial x_j} \right| \right) h_1$$

Assume that v' is a good approximation to $V'(x) = -\bar{c}$, then $\max\limits_{y \in Y_{od}} v'(y) \approx -\bar{c}$, where $\bar{c} > 0$ is a constant. Since the purpose of this estimate is to check if $v'(x) < 0,\ \forall x \in K$, then to show this we need

$$\frac{d}{2} \underbrace{\left(\max_{x \in K} \max_{j=1...d} \left| \frac{\partial v'(x)}{\partial x_j} \right| \right)}_{A_H} h_1 \leq \bar{c}$$

$$\Rightarrow h_1 \leq \frac{2\bar{c}}{d\ A_H}. \tag{5.33}$$

Thus, the number of points in $K = [-1, 1]^d$, i.e. $l = 1$, distributed in a square configuration

needs to be

$$\xrightarrow{\text{from(5.21)}} M = \left\lfloor \frac{2}{h_1} + 1 \right\rfloor^d,$$
$$\xrightarrow{\text{from(5.33)}} M \geq \left\lfloor \frac{2\ d\ A_H}{2\bar{c}} + 1 \right\rfloor^d \approx \left(\frac{2\ d\ A_H}{2\bar{c}} \right)^d = d^d \left(\frac{A_H}{\bar{c}} \right)^d. \tag{5.34}$$

Following the same steps, we obtain the number of points required to fill a set K with body centred square configuration.

In $\mathbb{R}^d$, the formula of the estimate is

$$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \frac{d}{4} \left(\max_{x \in K} \max_{j=1\ldots d} \left| \frac{\partial v'(x)}{\partial x_j} \right| \right) h_1.$$

This gives

$$\frac{d}{4} \underbrace{\left(\max_{x \in K} \max_{j=1\ldots d} \left| \frac{\partial v'(x)}{\partial x_j} \right| \right)}_{A_H} h_1 \leq \bar{c}.$$

$$\Rightarrow h_1 \leq \frac{4\bar{c}}{d\ A_H}. \tag{5.35}$$

Therefore, the number of body centred square points in $K = [-1, 1]^d$, i.e. $l = 1$, is given by

$$\xrightarrow{\text{from(5.22)}} M = \left\lfloor \frac{2}{h_1} + 1 \right\rfloor^d + \left\lfloor \frac{2}{h_1} \right\rfloor^d$$
$$\xrightarrow{\text{from(5.35)}} M \geq \left\lfloor \frac{2\ d\ A_H}{4\bar{c}} + 1 \right\rfloor^d + \left\lfloor \frac{2\ d\ A_H}{4\bar{c}} \right\rfloor^d \approx 2 \left(\frac{d\ A_H}{2\bar{c}} \right)^d$$
$$= 2 \left(\frac{d}{2} \right)^d \left(\frac{A_H}{\bar{c}} \right)^d$$
$$= \frac{1}{2^{d-1}}\ d^d \left(\frac{A_H}{\bar{c}} \right)^d. \tag{5.36}$$

Remark 5.4. *The final formulation of the first estimate with a body centred square configuration to be used in application is*

$$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \frac{d}{4} \left(\max_{x \in K} \max_{j=1\ldots d} \left| \frac{\partial v'(x)}{\partial x_j} \right| \right) h_1$$

The results of this study show that the combination of norms where $p = \infty$ and $q = 1$, along with the body centred square configuration has improved the estimate by removing the constants, caused by taking the maximum of the other norms, as well as reducing the number of points in the checking grid Y_{od} by a factor of $\frac{1}{2^{d-1}}$, where d is the dimension of the space.

5.3.2 The second estimate (5.4)

Factor (1): Different norms

Consider the term $\underbrace{\max_{z\in\mathcal{T}} \|H(z)\|_p \; h^2_{\|.\|_q}}_{B_2}$ in estimate (5.4). Remember that, due to Remark 5.1, we have established our estimate in terms of $\|H(z)\|_{\max}$, which is just a result of using the combination of norms where: $p = \infty$ and $q = 1$, as we will see in the proof of Proposition 5.2.

The following proposition is a modified version of Proposition 5.1, where the effect of changing the norms is clear on the formulation of the estimate.

Proposition 5.2. *Let $(x_0, x_1, \ldots, x_k) \in \mathbb{R}^d$ be affinely independent vectors, define $\mathcal{T} \coloneqq co\{x_0, x_1, \ldots, x_k\}$, such that x_0 is fixed and $h_{\|.\|_1} = \max_{j=0,\ldots,k} \|x_j - x_0\|_1$ is the largest distance from any vertex x_j to x_0 under the L_1-norm, and consider a convex combination $\sum_{i=0}^k \lambda_i x_i \in \mathcal{T}$. Let $U \subset \mathbb{R}^d$ be an open set with $\mathcal{T} \subset U$.*
Let $w \in C^2(U, \mathbb{R})$, then

$$\left| w\left(\sum_{i=0}^k \lambda_i x_i\right) - \sum_{i=0}^k \lambda_i w(x_i) \right| \leq B_S \; h^2_{\|.\|_1} = \left(\max_{z\in\mathcal{T}} \max_{r,s=1,\ldots,d} \left| \frac{\partial^2 w(z)}{\partial x_r \partial x_s} \right| \right) \; h^2_{\|.\|_1}. \tag{5.37}$$

where $B_S \coloneqq \|H(z)\|_{\max} = \left(\max_{z\in\mathcal{T}} \max_{r,s=1,\ldots,d} \left| \frac{\partial^2 w(z)}{\partial x_r \partial x_s} \right| \right)$, and $H(z)$ is the Hessian of w at z.

Proof. From Taylor's theorem

$$\begin{aligned} w\Big(\sum_{i=0}^k \lambda_i x_i\Big) &= w(x_0) + \nabla w(x_0) \sum_{i=0}^k \lambda_i (x_i - x_0)^T + \frac{1}{2} \sum_{i=0}^k \lambda_i (x_i - x_0) \; H(z) \sum_{j=0}^k \lambda_j (x_j - x_0) \\ &= \sum_{i=0}^k \lambda_i \Big(w(x_0) + \nabla w(x_0)(x_i - x_0) + \frac{1}{2}(x_i - x_0)^T H(z) \sum_{j=0}^k \lambda_j (x_j - x_0) \Big). \end{aligned}$$

for some z on the line segment between x_0 and $\sum_{i=0}^k \lambda_i x_i$.

Again, by Taylor's theorem we have for every $i = 0, 1, \ldots, k$ that

$$w(x_i) = w(x_0) + \nabla w(x_0)(x_i - x_0) + \frac{1}{2}(x_i - x_0)^T H(z_i)(x_i - x_0)$$

for some z_i on the line segment between x_0 and x_i. Hence

$$\begin{aligned}
\left| w\Big(\sum_{i=0}^{k}\lambda_i x_i\Big) - \sum_{i=0}^{k}\lambda_i w(x_i)\right| &= \frac{1}{2}\left|\sum_{i=0}^{k}\lambda_i (x_i-x_0)^T\Big(H(z)\sum_{j=0}^{k}\lambda_j(x_j-x_0) + H(z_i)(x_i-x_0)\Big)\right| \\
&\leq \frac{1}{2}\left\|\sum_{i=0}^{k}\lambda_i\,(x_i-x_0)\right\|_1\Big(\Big\|H(z)\sum_{j=0}^{k}\lambda_j(x_j-x_0)\Big\|_\infty \\
&\quad + \Big\|H(z_i)\,(x_i-x_0)\Big\|_\infty\Big) \\
&\overset{(*)}{\leq} \frac{1}{2}\left\|\sum_{i=0}^{k}\lambda_i\,(x_i-x_0)\right\|_1\Big(\|H(z)\|_{\max}\|x_j-x_0\|_1 \\
&\quad + \|H(z_i)\|_{\max}\|x_i-x_0\|_1\Big) \\
&\leq \frac{1}{2}\sum_{i=0}^{k}\lambda_i\, B_S\, \underbrace{\|x_i-x_0\|_1}_{\leq h_{\|.\|_1}}\Big(\underbrace{\|x_j-x_0\|_1}_{\leq h_{\|.\|_1}} + \underbrace{\|x_i-x_0\|_1}_{\leq h_{\|.\|_1}}\Big) \\
&\leq \frac{1}{2}\sum_{i=0}^{k}\lambda_i\, B_S\,(2\,h^2_{\|.\|_1}) \\
&= B_S\, h^2_{\|.\|_1},
\end{aligned}$$

where $B_S := \|H(z)\|_{\max} = \max\limits_{z\in\mathcal{T}}\max\limits_{r,s=1,\ldots,d}\left|\frac{\partial^2 w(z)}{\partial x_r \partial x_s}\right|$, and

$$\begin{aligned}
(*)\ \|Ax\|_\infty &= \max_{1\leq i\leq d}\left|\sum_{j=1}^{d} a_{ij}\ x_j\right|, \\
&\leq \max_{1\leq i\leq d}\Big(\sum_{j=1}^{d}|a_{ij}|\ |x_j|\Big), \\
&= \max_{1\leq i,j\leq d}|a_{ij}|\ .\ \sum_{j=1}^{d}|x_j|, \\
&= \|A\|_{\max}\ \|x\|_1.
\end{aligned}$$

□

Thus, using the combination of norms where $p=\infty$ and $q=1$, improves the original Proposition 5.1 by eliminating the constant d from the estimate, see Remark 5.1. Moreover, in Proposition 5.1, $h_{\|.\|_2} = diam(\mathcal{T}) = \max\|x_i - x_i\|_2,\ 0\leq i,j\leq k$, is the maximum distance between any two vertices in $\mathcal{T}$ under the L_2-norm, while in the modified version 5.2, $h_{\|.\|_1} = \max\|x_j - x_0\|_1$, is the maximum distance between any vertex in $\mathcal{T}$ to x_0 under

the L_1-norm.

In the next section, we calculate the value of $h_{\|.\|_1}$ for the two configurations of grid points by choosing x_0 in a particular way, which will in turn provide two versions of the estimate.

Factor (2): Distribution of grid points

As we mentioned before, this estimate requires a suitable triangulation T of the set K and $h_{\|.\|_1}$ depends on the distance between the vertex x_0 and the other vertices of the simplex $\mathcal{T} \in \mathrm{T}$. Therefore, for each distribution of grid points we need to find the optimal choice of the vertex x_0 such that the largest distance from any vertex of $\mathcal{T}$ to x_0, under the L_1-norm, is minimal.

The Square Configuration

A suitable triangulation for the square configuration is the standard one, which can be generated by Definition 5.4. But first we need to introduce some notations.

Remark 5.5. *[20] For the construction of our triangulation we use the set S_d of all permutations of the numbers $1, 2, \ldots, d$, the characteristic function $\mathcal{X}_{\mathcal{J}}(i)$ equal to one if $i \in \mathcal{J}$ and equal to zero if $i \notin \mathcal{J}$, and the standard orthonormal basis $\boldsymbol{e_1}, \boldsymbol{e_2}, \ldots, \boldsymbol{e_d}$ of $\mathbb{R}^d$. Further, we use the functions $\boldsymbol{R}^{\mathcal{J}} : \mathbb{R}^d \to \mathbb{R}^d$, defined for every $\mathcal{J} \subset \{1, 2, \ldots, d\}$ by*

$$\boldsymbol{R}^{\mathcal{J}}(x) \coloneqq \sum_{i=1}^{d} (-1)^{\mathcal{X}_{\mathcal{J}}(i)} x_i \boldsymbol{e}_i.$$

$\boldsymbol{R}^{\mathcal{J}}(x)$ puts a minus in front of the coordinates x_i of x whenever $i \in \mathcal{J}$.

Definition 5.4 (Standard triangulation)**.** *The triangulation* T *consists of the simplices*

$$\mathcal{T}_{z\mathcal{J}\sigma} \coloneqq co\{x_0^{z\mathcal{J}\sigma}, x_1^{z\mathcal{J}\sigma}, \ldots, x_d^{z\mathcal{J}\sigma}\}$$

scaled by h_1, for all $z \in \mathbb{N}_0^d$, all $\mathcal{J} \subset \{1, 2, \ldots, d\}$, and all $\sigma \in S_d$, where

$$x_i^{z\mathcal{J}\sigma} \coloneqq \boldsymbol{R}^{\mathcal{J}}\left(z + \sum_{j=1}^{i} \boldsymbol{e}_{\sigma(j)}\right) h_1 \quad \text{for} \quad i = 0, 1, 2, \ldots, d. \tag{5.38}$$

Figure 5.6 shows how the standard triangulation looks like for a square $[0, h_1]^2 \subset \mathbb{R}^2$ of a 2-D square configuration.

Our goal is to determine the optimal choice of the vertex x_0 in a simplex $\mathcal{T}$ of a square configuration, which will in turn give the largest possible distance from any vertex

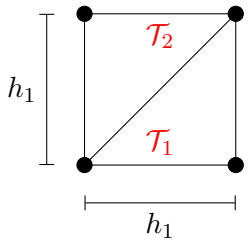


Figure 5.6: A square $[0, h_1]^2 \subset \mathbb{R}^2$ of a square configuration, where ($\bullet$) are our grid points, and $\mathcal{T}_i, i = 1, 2$ are simplices of the triangulation T.

$x_i \in \mathcal{T},\ i = 0, 1, \ldots, d$ to x_0. The following theorem gives the formula to determine x_0 as well as the value of $h_{\|.\|_1}$.

Theorem 5.7. *Let $\mathcal{T} \coloneqq co\{x_0, x_1, \ldots, x_d\}$ be a simplex of a triangulation* T *of a d-D square configuration of grid points. Since estimate (5.37) depends on the choice of x_0, then the optimal choice of the vertex x_0 of $\mathcal{T}$ is the middle point x_m, which is given by*

$$x_m \coloneqq \boldsymbol{R}^{\mathcal{J}} \left(z + \sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \right) h_1,\ \forall z \in \mathbb{N}_0^d. \tag{5.39}$$

and the maximum distance from any vertex $x_i, i = 1, 2, \ldots, d$ to x_0 in the L_1-norm is

$$h_{\|.\|_1} \coloneqq \max_{i=1,2,\ldots,d} \|x_i - x_0\|_1 = \left\lceil \frac{d}{2} \right\rceil h_1. \tag{5.40}$$

where $\lceil x \rceil$ is the ceiling function, (i.e., the least integer that is greater than or equal to x), and h_1 is the distance between the equally distanced grid points in all directions of the space.

Proof. The formula of the middle vertex follows directly from the construction equation (5.38) with $i = \lceil \frac{d}{2} \rceil$:

$$x_m = x_{\lceil \frac{d}{2} \rceil} \coloneqq \left(z + \sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \right) h_1, \tag{5.41}$$

where we set $\boldsymbol{R}^{\mathcal{J}} = id$, without loss of generality. Moreover, to find the largest distance

from x_m to the other vertices of $\mathcal{T}$ under the L_1-norm we write

$$\|x_i - x_m\|_1 = \left\| \left(z + \sum_{j=1}^{i} \boldsymbol{e}_{\sigma(j)} \right) h_1 - \left(z + \sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \right) h_1 \right\|_1, \ i = 0, 1, \ldots, d.$$

$$= \left\| \left(\sum_{j=1}^{i} \boldsymbol{e}_{\sigma(j)} - \sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \right) h_1 \right\|_1, \ i = 0, 1, \ldots, d$$

- if $i \geq \lceil \frac{d}{2} \rceil$ and d is even

$$\|x_i - x_m\|_1 = \left\| \left(\sum_{j=\lceil \frac{d}{2} \rceil + 1}^{i} \boldsymbol{e}_{\sigma(j)} \right) h_1 \right\|_1,$$

$$\leq \left\| \underbrace{\left(\boldsymbol{e}_{\sigma(\lceil \frac{d}{2} \rceil + 1)} + \ldots + \boldsymbol{e}_{\sigma(d)} \right)}_{\frac{d}{2}\text{-times}} h_1 \right\|_1,$$

$$= \frac{d}{2} h_1. \tag{5.42}$$

- if $i \geq \lceil \frac{d}{2} \rceil$ and d is odd

$$\|x_i - x_m\|_1 = \left\| \left(\sum_{j=\lceil \frac{d}{2} \rceil + 1}^{i} \boldsymbol{e}_{\sigma(j)} \right) h_1 \right\|_1,$$

$$\leq \left\| \underbrace{\left(\boldsymbol{e}_{\sigma(\lceil \frac{d}{2} \rceil + 1)} + \ldots + \boldsymbol{e}_{\sigma(d)} \right)}_{\frac{d-1}{2}\text{-times}} h_1 \right\|_1,$$

$$= \frac{d-1}{2} h_1. \tag{5.43}$$

From (5.42) and (5.43) we can write

$$\max_{i \geq \lceil \frac{d}{2} \rceil} \|x_i - x_m\|_1 = \left\lfloor \frac{d}{2} \right\rfloor h_1. \tag{5.44}$$

- if $i \leq \lceil \frac{d}{2} \rceil$ and d is even

$$\|x_m - x_i\|_1 = \left\| \left(\sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \right) h_1 \right\|_1,$$

$$\leq \left\| \underbrace{\left(\boldsymbol{e}_{\sigma(1)} + \ldots + \boldsymbol{e}_{\sigma(\lceil \frac{d}{2} \rceil)} \right)}_{\frac{d}{2}\text{-times}} h_1 \right\|_1,$$

$$= \frac{d}{2} h_1. \tag{5.45}$$

- if $i \leq \lceil \frac{d}{2} \rceil$ and d is odd

$$\begin{aligned} \|x_m - x_i\|_1 &= \left\| \Big(\sum_{j=1}^{\lceil \frac{d}{2} \rceil} \boldsymbol{e}_{\sigma(j)} \Big)\ h_1 \right\|_1, \\ &\leq \left\| \underbrace{\big(\boldsymbol{e}_{\sigma(1)} + \ldots + \boldsymbol{e}_{\sigma(\lceil \frac{d}{2} \rceil)} \big)}_{\frac{d+1}{2}\text{-times}}\ h_1 \right\|_1, \\ &= \frac{d+1}{2}\ h_1. \end{aligned} \tag{5.46}$$

From (5.45) and (5.46) we can write

$$\max_{i \leq \lceil \frac{d}{2} \rceil} \|x_m - x_i\|_1 = \left\lceil \frac{d}{2} \right\rceil\ h_1. \tag{5.47}$$

This shows that the maximum distance from any vertex $x_i \in \mathcal{T}$ to the middle point x_m under the L_1-norm is

$$\max_{i=1,\ldots,d} \|x_i - x_m\|_1 = \left\lceil \frac{d}{2} \right\rceil\ h_1.$$

□

Table 5.5 illustrate how to use Theorem 5.7 to determine the order of x_0 for even and odd dimensions.

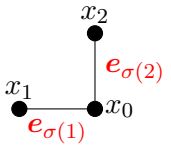
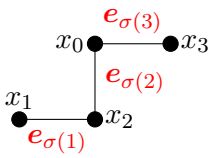
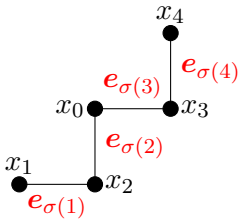
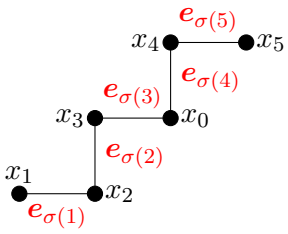
Dimension	Order of x_m	$h_{\|.\|_1} = \max\limits_{i=1,\dots,d} \|x_i - x_0\|_1$	x_0 in simplex $\mathcal{T}$
$d = 2$	$x_m := z + \sum_{j=1}^{\lceil \frac{2}{2} \rceil = 1} \boldsymbol{e}_{\sigma(j)}$	$\max\limits_{i=2} \|x_i - x_m\| = h_1$ $\max\limits_{i=1} \|x_m - x_i\| = h_1$	
$d = 3$	$x_m := z + \sum_{j=1}^{\lceil \frac{3}{2} \rceil = 2} \boldsymbol{e}_{\sigma(j)}$	$\max\limits_{i=3} \|x_i - x_m\| = h_1$ $\max\limits_{i=1,2} \|x_m - x_i\| = 2\ h_1$	
$d = 4$	$x_m := z + \sum_{j=1}^{\lceil \frac{4}{2} \rceil = 2} \boldsymbol{e}_{\sigma(j)}$	$\max\limits_{i=3,4} \|x_i - x_m\| = 2\ h_1$ $\max\limits_{i=1,2} \|x_m - x_i\| = 2\ h_1$	
$d = 5$	$x_m := z + \sum_{j=1}^{\lceil \frac{5}{2} \rceil = 3} \boldsymbol{e}_{\sigma(j)}$	$\max\limits_{i=4,5} \|x_i - x_m\| = 2\ h_1$ $\max\limits_{i=1,2,3} \|x_m - x_i\| = 3\ h_1$	

Table 5.5: The application of Theorem 5.7 to determine the optimal order of the vertex x_0 of a simplex $\mathcal{T}$ in $d = 2, 3, 4, 5$.

The body centred square configuration

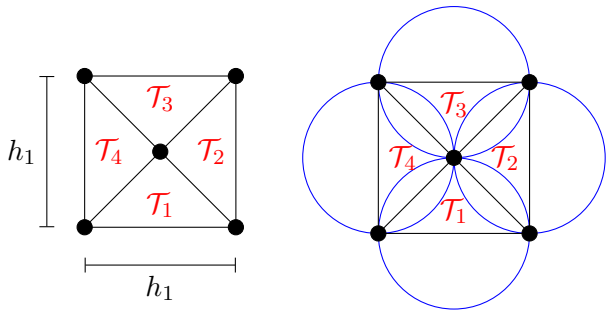
A suitable triangulation for the body centred square structure of grid points in Y_{od} is the Delaunay triangulation DT. As we mentioned before, we only consider a hypercube $\overline{S} = [0, h_1]^d$ of a d-D body centred square configuration, as the results obtained will be applied to the whole point set by symmetry.

Definition 5.5. *(The empty circle property for triangles [16, Definition 6.8]). A triangu-*

lation of a finite point set $P \subset \mathbb{R}^2$ is called a Delaunay triangulation, if the circumcircle of every triangle, the unique circle passing through the three vertices of the triangle, is empty, that is there is no point from P in its interior.

Let $\overline{S} = [0, h_1]^2 \subset \mathbb{R}^2$ be a square of a 2-D body centred square configuration. Obviously, the vertices of $\overline{S}$ are not collinear, thus it has a triangulation T [16, Proposition 6.2]. Since the empty circle property holds for all simplices $\mathcal{T}_i,\ i = 1, 2, 3, 4$ in the triangulation T of $\overline{S}$, hence the triangulation T is a Delaunay triangulation $DT(\overline{S})$ [29, Corollary 3.1], see Figure 5.7.

Similarly, a triangulation T for a hypercube $\overline{S} = [0, h_1]^d \subset \mathbb{R}^d$ is a Delaunay triangulation if the circum-hypersphere of any simplex in the $DT(\overline{S})$ is empty [23].



(a) The Delaunay triangulation.

(b) The empty circle property.

Figure 5.7: (a) The Delaunay triangulation of the vertices (•) of a square $\overline{S} = [0, h_1]^2 \subset \mathbb{R}^2$ of a body centred square configuration, where h_1 is the distance between grid points in both directions, (b) All triangles $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$, and $\mathcal{T}_4$ satisfy the empty circle property.

Theorem 5.8. *Let $\mathcal{T} := co\{x_0, x_1, \ldots, x_d\}$ be a simplex of the Delaunay triangulation DT of a d-D body centred square configuration of grid points. Then, the optimal choice of the vertex x_0 of $\mathcal{T}$ is the centre grid point x_m*

$$x_m := \underbrace{\left(\frac{h_1}{2}, \frac{h_1}{2}, \ldots, \frac{h_1}{2}\right)}_{d\text{-}times},$$

and the maximum distance from any vertex $x_i \in \mathcal{T}, i = 1, 2, \ldots, d$ to x_0 under the L_1-norm is

$$h := \max_{i=1,2,\ldots,d} \|x_i - x_0\|_1 = \frac{d}{2}\, h_1,$$

where h_1 is the distance between the equally distanced grid points in all directions of the space.

Proof. Let $\overline{S} = [0, h_1]^d \subset \mathbb{R}^d$ be a hypercube of a d-D body centred square configuration of grid points, and let $\mathcal{T} = co\{x_0, x_1, \ldots, x_d\} \in DT$ be a simplex of the Delaunay triangulation of $\overline{S}$.

Thus, the optimal choice of the vertex $x_0 \in \mathcal{T}$, that minimizes the largest distance between the vertices of $\mathcal{T}$, is the centre grid point $x_m := \underbrace{\left(\frac{h_1}{2}, \frac{h_1}{2}, \ldots, \frac{h_1}{2}\right)}_{\text{d-times}}$, since it has equal distances to all the other vertices of $\mathcal{T}$. This means

$$\begin{aligned} \max_{i=1,2,\ldots,d} \|x_i - x_m\|_1 &= \sum_{i=1}^{d} \left|x_i - \frac{h_1}{2}\right|, \\ &= \left|x_1 - \frac{h_1}{2}\right| + \left|x_2 - \frac{h_1}{2}\right| + \ldots + \left|x_d - \frac{h_1}{2}\right|. \end{aligned} \tag{5.48}$$

Let x_i be a vertex of $\mathcal{T}$, then $(x_i)^j \in \{0, h_1\}$ and thus in all cases

$$\begin{aligned} \max_{i=1,2,\ldots,d} \|x_i - x_m\|_1 &= \underbrace{\left|-\frac{h_1}{2}\right| + \left|-\frac{h_1}{2}\right| + \ldots + \left|-\frac{h_1}{2}\right|}_{\text{d-times}}, \\ &= \frac{d}{2}\, h_1, \end{aligned} \tag{5.49}$$

and any other two vertices of $\mathcal{T}$ have a distance of at least h_1. □

The improved formula of the second estimate:

In this section, we state the improved formula of the second estimate (5.4), after combining the effects of the two factors on its formulation. Table 5.6 and Table 5.7 present the two versions of the estimate according to the used structure of the checking points Y_{od}.

The formulas (5.50) and (5.51), show that the estimates for both configurations is the same for even-dimensional spaces. However, for odd-dimensional spaces, the body centred square configurations provides a sharper estimate.

Best configuration for this estimate:

We are going to finish this investigation by checking the distribution of grid points in Y_{od} which requires fewer grid points.

$\max_{i=1,2,\ldots,d} \|x_i - x_0\|$	d-D Square grid
$h_{\|.\|_1} = \left\lceil \frac{d}{2} \right\rceil h_1$	$v'(\sum_{i=1}^{k} \lambda_i x_i) \leq \max_{y \in Y_{od}} v'(y) + \left(\left\lceil \frac{d}{2} \right\rceil \right)^2 \left(\max_{z \in \mathcal{T}} \max_{r,s=1,\ldots,d} \left\lvert \frac{\partial^2 v'(z)}{\partial x_r \partial x_s} \right\rvert \right) h_1^2$ (5.50)

Table 5.6: The improved formula of the estimate for a d-D square grid, with $h_{\|.\|_1}$ being the maximum distance from any vertex x_i of a simplex $\mathcal{T}$ to the vertex x_0 under the L_1-norm.

$\max_{i=1,2,\ldots,d} \|x_i - x_0\|$	d-D Body centred square grid
$h_{\|.\|_1} = \frac{d}{2} h_1$	$v'(\sum_{i=1}^{k} \lambda_i x_i) \leq \max_{y \in Y_{od}} v'(y) + (\frac{d}{2})^2 \left(\max_{z \in \mathcal{T}} \max_{r,s=1,\ldots,d} \left\lvert \frac{\partial^2 v'(z)}{\partial x_r \partial x_s} \right\rvert \right) h_1^2$ (5.51)

Table 5.7: The improved formula of the estimate for a d-D body centred square grid, with $h_{\|.\|_1}$ being the maximum distance from any vertex x_i of a simplex $\mathcal{T}$ to the vertex x_0 under the L_1-norm.

Proposition 5.3. *Consider a hypercube $K = [-1, 1]^d \subset \mathbb{R}^d$, filled with points of the checking grid Y_{od}, then*

- *If d is even, the best distribution of the points in K, which requires fewer points, is the square configuration.*
- *If d is odd, the best distribution of the points in K, which requires fewer points, is the body centred square configuration.*

Proof. Recall the formula of the estimate for the d-D square configuration

$$v'\left(\sum_{i=1}^{k} \lambda_i x_i\right) \leq \max_{y \in Y_{od}} v'(y) + \left(\left\lceil \frac{d}{2} \right\rceil \right)^2 \left(\max_{z \in \mathcal{T}} \max_{r,s=1,\ldots,d} \left\lvert \frac{\partial^2 v'(z)}{\partial x_r \partial x_s} \right\rvert \right) h_1^2$$

If d is even, then this formula will take the form

$$v'\left(\sum_{i=1}^{k} \lambda_i x_i\right) \leq \max_{y \in Y_{od}} v'(y) + \left(\frac{d}{2} \right)^2 \left(\max_{z \in \mathcal{T}} \max_{r,s=1,\ldots,d} \left\lvert \frac{\partial^2 v'(z)}{\partial x_r \partial x_s} \right\rvert \right) h_1^2$$

Assume that v' is a good approximation to $V'(x) = -\bar{c}$, then $\max_{y \in Y_{od}} v'(y) \approx -\bar{c}$, where $\bar{c} > 0$ is a constant. Consequently

$$\left(\frac{d}{2} \right)^2 \underbrace{\left(\max_{z \in \mathcal{T}} \max_{r,s=1,\ldots,d} \left\lvert \frac{\partial^2 v'(z)}{\partial x_r \partial x_s} \right\rvert \right)}_{B_H} h_1^2 \leq \bar{c}$$

$$\Rightarrow h_1^2 \leq \frac{\bar{c}}{B_H}\left(\frac{2}{d}\right)^2,$$
$$\Rightarrow h_1 \leq \sqrt{\frac{\bar{c}}{B_H}}\,\frac{2}{d}. \tag{5.52}$$

Let $K = [-1,1]^d \subset Y_{od}$, be a set filled with checking grid points distributed in a square from, then the required number of points in K is given by

$$\xRightarrow{\text{from (5.21)}} M = \left\lfloor \frac{2}{h_1} + 1 \right\rfloor^d$$
$$\xRightarrow{\text{from (5.52)}} M \geq \left\lfloor 2\,(\frac{d}{2})\sqrt{\frac{B_H}{\bar{c}}} + 1 \right\rfloor^d \approx \left(d\sqrt{\frac{B_H}{\bar{c}}}\right)^d$$
$$\Rightarrow M \geq \big(d\big)^d \Big(\frac{B_H}{\bar{c}}\Big)^{\frac{d}{2}}. \tag{5.53}$$

While if d is odd, then the estimate will look like

$$v'\left(\sum_{i=1}^{k}\lambda_i x_i\right) \leq \max_{y\in Y_{od}} v'(y) + \left(\frac{d+1}{2}\right)^2 \left(\max_{z\in\mathcal{T}}\max_{r,s=1,\ldots,d}\Big|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\Big|\right) h_1^2$$

then

$$\left(\frac{d+1}{2}\right)^2 \underbrace{\left(\max_{z\in\mathcal{T}}\max_{r,s=1,\ldots,d}\Big|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\Big|\right)}_{B_H} h_1^2 \leq \bar{c}$$

$$\Rightarrow h_1^2 \leq \frac{\bar{c}}{B_H}\left(\frac{2}{d+1}\right)^2,$$
$$\Rightarrow h_1 \leq \sqrt{\frac{\bar{c}}{B_H}}\left(\frac{2}{d+1}\right). \tag{5.54}$$

And the number of grid points in K will be

$$\xRightarrow{\text{from (5.21)}} M = \left\lfloor \frac{2}{h_1} + 1 \right\rfloor^d$$
$$\xRightarrow{\text{from (5.52)}} M \geq \left\lfloor 2\,(\frac{d+1}{2})\sqrt{\frac{B_H}{\bar{c}}} + 1 \right\rfloor^d \approx (d+1)\sqrt{\frac{B_H}{\bar{c}}})^d$$
$$\Rightarrow M \geq \big(d+1\big)^d \Big(\frac{B_H}{\bar{c}}\Big)^{\frac{d}{2}}. \tag{5.55}$$

On the other hand, for the d-D body centred square configuration our estimate has the form

$$v'(\sum_{i=1}^{k}\lambda_i x_i) \leq \max_{y\in Y_{od}} v'(y) + \left(\frac{d}{2}\right)^2 \left(\max_{z\in\mathcal{T}}\max_{r,s=1,\ldots,d}\Big|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\Big|\right) h_1^2$$

which is the same for odd and even dimensions. Thus

$$\left(\frac{d}{2}\right)^2 \underbrace{\left(\max_{z\in\mathcal{T}} \max_{r,s=1,\ldots,d} \left|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\right|\right)}_{B_H} h_1^2 \leq \bar{c},$$

$$\Rightarrow h_1^2 \leq \frac{\bar{c}}{B_H} \left(\frac{2}{d}\right)^2,$$
$$\Rightarrow h_1 \leq \sqrt{\frac{\bar{c}}{B_H}}\, \frac{2}{d}. \tag{5.56}$$

Assume K is now filled with checking grid points distributed in a body centred square form, then the number of points which K needs is

$$\xRightarrow{\text{from (5.22)}} M = \left\lfloor \frac{2}{h_1} + 1 \right\rfloor^d + \left\lfloor \frac{2}{h_1} \right\rfloor^d$$
$$\xRightarrow{\text{from (5.56)}} M \geq \left\lfloor 2\,(\frac{d}{2})\,\sqrt{\frac{B_H}{\bar{c}}} + 1 \right\rfloor^d + \left\lfloor 2\,(\frac{d}{2})\,\sqrt{\frac{B_H}{\bar{c}}} \right\rfloor^d \approx 2\,\left(d\,\sqrt{\frac{B_H}{\bar{c}}}\right)^d$$
$$\Rightarrow M \geq 2\,\left(d\right)^d \left(\frac{B_H}{\bar{c}}\right)^{\frac{d}{2}}. \tag{5.57}$$

This shows that when d is even, then the square configuration requires less points than the body centred square one, in contrast to the odd dimensional cases. □

The proposed verification estimates will add a significant improvement to the construction method of Lyapunov functions using Radial Basis Functions in the sense that a constructed function with either a regular grid or with the refinement algorithms can be proved to be indeed a Lyapunov function. A remarkable finding of these estimates is shown in the following theorem.

Theorem 5.9. *The following algorithm for the verification estimates for the constructed function with the RBF method and a regular grid of collocation points will always terminate and successfully construct and verify a Lyapunov function on a compact set* $K \subseteq A(x_0)\backslash E_{nh}$ *if the grid is chosen in the following way*

- *in the k-th step*

$$h_{RBF} \leq 10^{-k} \;\Rightarrow\; h_1 \leq \frac{10^{-k}}{\boldsymbol{\beta}_k} \Big\backslash h_1 \leq \frac{10^{-k}}{\sqrt{\boldsymbol{\beta}_k}}. \tag{5.58}$$

where h_{RBF} is the fill distance of X_{RBF}, h_1 is the density of the checking grid Y_{od}, E_{nh} is a small neighbourhood around the equilibrium point, and $\boldsymbol{\beta}_k$ is derived from the solution of the linear system solved at the step k.

Proof. Let v be an RBF approximant constructed by approximating the solution of one of the Lyapunov functions satisfying $V'(x) = -\|x - x_0\|^2$, or $V'(x) = -\bar{c}, \bar{c} > 0$, using the RBF method. Then, according to the theoretical error estimate on the orbital derivative (2.12), the RBF approximant v is a Lyapunov function, i.e., $v'(x) < 0,\ \forall x \in K$, if the fill distance h_{RBF} is fine enough. Moreover, assume that $h_{RBF} \leq 10^{-k}$ is the fill distance for the RBF grid X_{RBF}, where $k = 1, 2, \ldots$, indicates the steps. Now, by making k larger and larger we will definitely reach a step k_1, where h_{RBF} is small enough, such that v' is a good approximation to V' and thus

$$\max_{y \in Y_{od}} v'(y) \leq -\epsilon,\ \epsilon > 0,\ Y_{od} \subset K. \qquad \forall k \geq k_1$$

Using the verification estimates to show that $v'(x) \leq -\epsilon,\ \forall x \in K$, (i.e., strictly negative), yields:

The first estimate:

$$v'(x) \leq \underbrace{\max_{y \in Y_{od}} v'(y)}_{\leq -\epsilon} + \underbrace{(\boldsymbol{\beta}_k\ C_1)\ h_1}_{\leq \frac{\epsilon}{2}} \leq -\frac{\epsilon}{2} \tag{5.59}$$

where $\boldsymbol{\beta}_k$ is the solution of the linear system produced by the RBF approximation on a grid X_{RBF} of size $h_{RBF} = 10^{-k}$. We obtain (5.59) if

$$\boldsymbol{\beta}_k\ C_1\ h_1 \leq \frac{\epsilon}{2} \Leftrightarrow h_1 \leq \frac{\epsilon}{2\ \boldsymbol{\beta}_k\ C_1}. \tag{5.60}$$

The second estimate:

$$v'(\sum_{i=1}^{k} \lambda_i x_i) \leq \underbrace{\max_{y \in Y_{od}} v'(y)}_{\leq -\epsilon} + \underbrace{(\boldsymbol{\beta}_k\ C_2)\ h_1^2}_{\leq \frac{\epsilon}{2}}. \tag{5.61}$$

and we obtain (5.61) if

$$\boldsymbol{\beta}_k\ C_2\ h_1^2 \leq \frac{\epsilon}{2} \Leftrightarrow h_1 \leq \sqrt{\frac{\epsilon}{2\ \boldsymbol{\beta}_k\ C_2}}. \tag{5.62}$$

The constants ϵ, C_1 and C_2 are unknown but at some point

$$10^{-k} \leq \frac{\epsilon}{2C_1} \quad \text{and} \quad 10^{-k} \leq \sqrt{\frac{\epsilon}{2C_2}}, \qquad \forall\ k \geq k_2.$$

Thus, by (5.58) the estimates hold for $k \geq K$ where $K = \max(k_1, k_2)$. □

5.4 Application to Numerical Examples

This section illustrates how to apply these verification estimates to check if the RBF approximant v is a Lyapunov function. More precisely, the main objective of the section is to show that $v'(x) \leq 0$, $\forall\, x \in K$. Thus, we are going to consider the numerical examples where we have constructed approximants v with *"regular"* grid of collocation points.
The steps of the verification procedure are:

1. Fix a compact and convex set $K \subset \mathbb{R}^d$ as well as the two subsets: $X_{RBF} \subseteq K$, for the construction method, and $Y_{od} \subseteq K$, for the checking in which we either distribute the points in a square or body centred square configurations.
2. Fix a radial basis function $\Psi(x) = \psi(\|x\|)$.
3. Approximate one of the Lyapunov functions $V = Q$ or $V = T$, using the RBF construction method.
4. Calculate the quantities F, D_1, D_2, and $\boldsymbol{\beta}$, then substitute them in the formulas of the first and second derivatives of the orbital derivative (5.9) and (5.15), respectively.
5. Use the estimates formula to obtain a rule of thumb for the order of h_1, i.e., the density of the checking grid Y_{od}.
6. Calculate the sign and value of v' over the checking grid of density h_1, then determine $\max\limits_{y \in Y_{od}} v'(y)$.
7. Finally, we use the estimates formula again to show $v'(x) < 0$ for all $x \in K$, if yes then the constructed function v is a Lyapunov function.
8. Otherwise, if 7 fails, then we might need a smaller X_{RBF} grid or a smaller checking grid Y_{od}.

Example 5.1. *Consider the non-linear system*

$$\begin{cases} \dot{x} = -x - 2y + x^3, \\ \dot{y} = -y + \frac{1}{2}x^2 y + x^3. \end{cases}$$

1. *We have fixed a compact set* $K = [-1, 1]^2 \subset \mathbb{R}^2$, *and*
   a regular grid $X_{RBF} = \{(x, y) \in \mathbb{R}^2 \mid (x, y) \in \{0, \pm h, \pm 2h, \ldots, \pm 1\}\} \backslash \{(0, 0)\}$, *with* $h = 0.11 \Rightarrow N = 360$ *points.*

2. *We have chosen the Wendland basis function* $\psi_{6,4}(c\|x\|)$ *with* $c = 1$.

3. *Then we have approximated the Lyapunov function satisfying* $V'(x) = -\|x\|^2$, *and by solving the resulting linear system we got* $\boldsymbol{\beta} = \sum_{k=1}^{360} |\beta_k| = 15.9078$.

4. *After that we have calculated the quantities:*

   $F \coloneqq \max\limits_{x\in K} \|f(x)\|_2 = 4.717,$
   $D_1 \coloneqq \max\limits_{x\in K} \max\limits_{j=1,\dots,N} \|\frac{\partial f(x)}{\partial x_j}\|_2 = 5.657,$
   $D_2 \coloneqq \max\limits_{x\in K} \max\limits_{i,j=1,\dots,N} \|\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\|_2 = 9.2195.$
   (For the detailed calculations see the Appendix).

5. *Now we apply our estimates to determine the value of* h_1*:*

   - *The first estimate:*

     Recall the formula of the first derivative of the orbital derivative (5.9)

     $$\left|\frac{\partial v'(x)}{\partial x_j}\right| \leq \boldsymbol{\beta}\Big(9679.37\ c^3F^2 + 1124.77\ c^2FD_1\Big).$$

     Then substitute for the values of c, F, D_1, $\boldsymbol{\beta}$ *and get*

     $$\begin{aligned}\left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(9679.37\ (4.717)^2 + 1124.77\ (4.717)\ (5.657)\Big),\\ &= 15.9078\ \Big(215366.844 + 30013.440\Big)\\ &\leq 3.9 \times 10^6. \end{aligned} \tag{5.63}$$

     Remember that, for this estimate, the best distribution of points in the checking grid Y_{od} *is the body centred square configuration. Therefore, to determine the density of* Y_{od}, *we use the formula of the estimate*

     $$\begin{aligned} v'(x) &\leq \max_{y\in Y_{od}} v'(y) + \frac{1}{2}\ (\max_{x\in K}\max_{j=1,2} |\frac{\partial v'(x)}{\partial x_j}|)\ h_1,\\ &= \max_{y\in Y_{od}} v'(y) + \frac{1}{2}\ \Big(3.9 \times 10^6\Big)\ h_1,\\ &= \max_{y\in Y_{od}} v'(y) + 1.95 \times 10^6\ h_1 \end{aligned} \tag{5.64}$$

     Assuming that the orbital derivative of constructed function v' *is a good approximation to* $V' \Rightarrow \max\limits_{y\in Y_{od}} v'(y) \approx -\epsilon$, *which yields*

     $$1.95 \times 10^6\ h_1 \leq \epsilon\ \Rightarrow h_1 \leq \frac{\epsilon}{1.95}\ 10^{-6}. \tag{5.65}$$

Finally, to show that the approximant v is a Lyapunov function, we check the value and the sign of v' over Y_{od} with $h_1 \leq 10^{-6}$. If $\max_{y \in Y_{od}} v'(y) + 1.95 \times 10^6 \; h_1 \leq 0$, then (5.64) yields $v'(x) \leq 0, \; \forall \; x \in K$ and thus v is a Lyapunov function. Otherwise, we will need to construct an RBF approximant with smaller X_{RBF} and reapply the verification process. Note that the value of h_1 is too small, thus requires a long calculation time. However, this will be improved with the second estimate.

- *The second estimate:*

  We start by recalling the formula of the second derivative of the orbital derivative (5.15)

  $$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq \boldsymbol{\beta}\big(157870.623 \; c^4 F^2 + 19358.745 \; c^3 F D_1 + 1124.797 \; c^2 F D_2\big).$$

  then substitute for the values of c, F, D_1, D_2, $\boldsymbol{\beta}$ and get

  $$\begin{aligned}\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq 15.9078 \; \Big(157870.623 \; (4.717)^2 + 19358.745 \; (4.717) \; (5.657) + 1124.797 \\ &\quad (4.717) \; (9.2195)\Big), \\ &\leq \Big(6.5 \times 10^7\Big). \end{aligned} \tag{5.66}$$

  For this estimate, the best distribution of points in Y_{od} is the square configuration. Thus, the corresponding formula of the estimate is

  $$\begin{aligned} v'\left(\sum_{i=1}^{k} \lambda_i x_i\right) &\leq \max_{y \in Y_{od}} v'(y) + \left(\max_{z \in \mathcal{T}} \max_{r,s=1,2} \left|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\right|\right) \; h_1^2, \\ &= \max_{y \in Y_{od}} v'(y) + \Big(6.5 \times 10^7\Big) \; h_1^2. \end{aligned} \tag{5.67}$$

  Assume that v' is a good approximant to V', then $\max_{y \in Y_{od}} v'(y) \approx -\epsilon$, which yields

  $$6.5 \times 10^7 \; h_1^2 \leq \epsilon \; \Rightarrow h_1 \leq \sqrt{\frac{\epsilon}{6.5}} \; 10^{-3.5} \tag{5.68}$$

  Now, we are going to verify the negativity of the orbital derivative over Y_{od} with density $h_1 \leq 10^{-3}$. Thus, we choose $h_1 = 0.3125 \; \big(10^{-4}\big)$, then calculating the sign and value of v' gives $\max_{y \in Y_{od}} v'(y) = -0.0825$. Consequently, (5.67) shows that

  $$\begin{aligned} v'(x) &\leq -0.0825 + \big(6.5 \times 10^7\big) \; \big(0.3125 \times 10^{-4}\big)^2, \\ &= -0.0825 + 0.063 = -0.0195. \end{aligned}$$

Thus, $v'(x) < 0$, $\forall\, x \in K\backslash E_{nh}$, where $E_{nh} = [-0.1, 0.1]^2$, i.e., this RBF approximant v is a Lyapunov function.

Comparing the output of the first and the second estimates, (5.65) and (5.68) respectively, shows that the second one is better in terms of the value of h_1. More precisely, the first estimate requires a smaller h_1, thus needs longer calculation time for the verification process.

Example 5.2. *Consider the non-linear system [17, Example 2.10]*

$$\begin{cases} \dot{x} = -x(1 - x^2 - y^2) - y, \\ \dot{y} = -y(1 - x^2 - y^2) + x. \end{cases}$$

1. *For this system we have fixed a compact set $K = [-0.9, 0.9]^2 \subset \mathbb{R}^2$, and a regular grid $X_{RBF} = \{(x, y) \in \mathbb{R}^2 \mid (x, y) \in \{0, \pm h, \pm 2h, \ldots, \pm 0.9\}\}\backslash\{(0, 0)\}$, with $h = 0.15 \Rightarrow N = 168$ collocation points.*
2. *We chose the Wendland basis function $\psi_{6,4}(c\|x\|)$ with $c = 1$.*
3. *We have used the RBF approximation method to approximate the Lyapunov function satisfying $V'(x) = -1$, then by solving the resulting linear system we obtained $\boldsymbol{\beta} = \sum_{k=1}^{168} |\beta_k| = 8.54$.*
4. *Calculate the quantities: $F = 3.57$, $D_1 = 4.98$, and $D_2 = 5.69$. (For the detailed calculations see Appendix.)*
5. *Use the estimates formula to find the value of h_1.*

   - *The first estimate:*

     First, we substitute for the values of c, F, D_1 and $\boldsymbol{\beta}$ in the formula of the first derivative of the orbital derivative (5.9):

$$\begin{aligned} \left|\frac{\partial v'(x)}{\partial x_j}\right| &\leq \boldsymbol{\beta}\Big(9679.37\, c^3F^2 + 1124.77\, c^2FD_1\Big), \\ &\leq 8.54\, \Big(9679.37\, (3.57)^2 + 1124.77\, (3.57)\, (4.98)\Big), \\ &= \Big(1.2 \times 10^6\Big). \end{aligned} \tag{5.69}$$

     Then, we determine the density of Y_{od} using the formula of the first estimate for

the body centred square configuration

$$\begin{aligned} v'(x) &\leq \max_{y\in Y_{od}} v'(y) + \frac{1}{2}\left(\max_{x\in K}\max_{j=1,2}\left|\frac{\partial v'(x)}{\partial x_j}\right|\right) h_1, \\ &= \max_{y\in Y_{od}} v'(y) + \frac{1}{2}\left(1.2\times 10^6\right) h_1, \\ &= \max_{y\in Y_{od}} v'(y) + 0.6\times 10^6\ h_1 \end{aligned} \tag{5.70}$$

If we assume that the orbital derivative of the approximant v is a good approximation to $V' \Rightarrow \max\limits_{y\in Y_{od}} v'(y) \approx -\epsilon$. Therefore, to show that $v'(x) \leq 0,\ \forall\ xinK$, we need

$$0.6\times 10^6\ h_1 \leq \epsilon\ \Rightarrow h_1 \leq \frac{\epsilon}{0.6}\ 10^{-6}. \tag{5.71}$$

Finally, to verify the negativity of the orbital derivative of v, we calculate the sign and value of v' over the checking grid Y_{od} with density $h_1 \leq 10^{-6}$. If $\max\limits_{y\in Y_{od}} v'(y) + 0.6\times 10^6\ h_1 < 0$, then by (5.70) the RBF approximant is a Lyapunov function. However, if not we need a smaller X_{RBF} to construct v.

The second estimate:

Recall the formula of the second derivative of the orbital derivative (5.15), then substitute for c, F, D_1, D_2, and $\boldsymbol{\beta}$:

$$\begin{aligned} \left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq \boldsymbol{\beta}\big(157870.623\ c^4F^2 + 19358.745\ c^3FD_1 + 1124.797\ c^2FD_2\big), \\ &\leq 8.54\ \Big(157870.623\ (3.57)^2 + 19358.745\ (3.57)\ (4.98) + 1124.797 \\ &\quad (3.57)\ (5.69)\Big), \\ &= \Big(2.03\times 10^7\Big). \end{aligned} \tag{5.72}$$

For this estimate, the best distribution of points in Y_{od} is the square configuration. Thus, the corresponding formula of the estimate is

$$\begin{aligned} v'\left(\sum_{i=1}^{k}\lambda_i x_i\right) &\leq \max_{y\in Y_{od}} v'(y) + \left(\max_{z\in\mathcal{T}}\max_{r,s=1,2}\left|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\right|\right)\ h_1^2, \\ &= \max_{y\in Y_{od}} v'(y) + \left(2.03\times 10^7\right) h_1^2. \end{aligned} \tag{5.73}$$

Then, to show $v'(x) \leq 0, \forall\ x \in K$, we assume that v' is a good approximation to $V' \Rightarrow \max\limits_{y\in Y_{od}} v'(y) = -\epsilon$, with ϵ close to 1. Thus,

$$2.03\times 10^7\ h_1^2 \leq \epsilon\ \Rightarrow h_1 \leq \sqrt{\frac{\epsilon}{2.03}}\ 10^{-3.5} \tag{5.74}$$

For the verification process, we choose $h_1 = 0.9\ \left(10^{-4}\right)$*, then check the sign and value of* v' *over* Y_{od} *which gives* $\max_{y \in Y_{od}} v'(y) = -0.395$*. As a result, form (5.73) we get*

$$\begin{aligned} v'(x) &\leq -0.395 + \left(2.03 \times 10^7\right)\ \left(0.9 \times 10^{-4}\right)^2, \\ &= -0.395 + 0.164 = -0.231. \end{aligned}$$

This shows that $v'(x) < 0,\ \forall\ x \in K \backslash E_{nh}$*, where* $E_{nh} = [-0.1, 0.1]^2$*. Thus the constructed RBF approximant is a Lyapunov function.*

In this Chapter, we have designed, analysed, and implemented two algorithms for the sake of providing a more reliable and accurate estimation of the density of the checking grid, where we verify the negativity of the orbital derivative of an RBF approximant. The calculations of both estimates are straightforward and most of the quantities are computed by hand except the value of $\boldsymbol{\beta}$ and the final checking of the sign of the orbital derivative over Y_{od} with the estimated density h_1. However, the second verification estimate has showed a better results as it decreases the size of h_1 by nearly a half, which improves the computation time significantly.

Chapter 6

Combining Refinement and Verification

In the previous Chapter, we have derived two verification estimates to verify the negativity of the orbital derivative of a constructed function using the RBF method. So far, these estimates were examined on the RBF approximants constructed on a regular grid. However, a successful construction of RBF approximants was obtained by using the original grid refinement and the modified grid refinement algorithms, as shown in Chapters 3 and 4. Therefore, the motivation of *the combination method* is to apply the verification estimates to the functions v constructed with one of the refinement algorithms to check if $v'(x) < 0$ for all $x \in K$.

6.1 The steps of the combination method

Let v be an RBF approximant constructed with the refinement algorithm, such that $X_{final} \subset K$ is the set of grid points obtained after the refinement process, and $Y_{od} \subset K$ denotes the checking grid structured in a square or body centred square forms. Then, the steps of the combination method are:

1. Calculate the quantities F, D_1, D_2 and $\boldsymbol{\beta}$, and substitute in the formula of the first estimate (5.32) or the second estimate (5.50) if d is even, and (5.51) if d is odd.
2. Use the estimates to determine the order of h_1, i.e., the density of the checking grid

Y_{od}.

3. Use this h_1 to check the sign of the orbital derivative of the RBF approximant v over Y_{od} as well as to calculate the maximum value of the orbital derivative.
4. Again, we use the estimates formula to show that $v'(x) \leq 0$, for all $x \in K$, and thus the constructed function v is a Lyapunov function.
5. If 4 fails, i.e. $v'(x) > 0$ for some $x \in K$, then the initial grid of the refinement or the checking grid is not fine enough.

6.2 Numerical Examples

6.2.1 Examples solved with the refinement algorithm

In this section we apply the combination method to the examples solved in Chapter 3, where we have constructed RBF approximants using the refinement algorithm.
Note that, we will skip the first estimate as it needs a smaller h_1 than the second one, thus more computation time.

Example 6.1. *Consider the system in Example 3.2.*
For this example, we have used the refinement algorithm to construct an approximant v of the Lyapunov function satisfying $V'(x) = -\|x\|^2$. We started the refinement process with an initial grid $X_{initial} = 24$ points and were able to construct a function v with $X_{final} = 88$ points. Moreover, we have checked if the sign of v' is negative over a checking grid X_{check} of size $h_{check} = 10^{-3}$. However, the combination method will provide a different estimate for the size of the checking grid, to verify the negativity of the orbital derivative of v.

1. *Recall that: $F = 4.717$, $D_1 = 5.657$, and $D_2 = 9.2195$.*
2. *We calculate the value of $\boldsymbol{\beta} = \sum_{k=1}^{88} |\beta_k| = 0.4105$.*
3. *Use the verification estimates to determine the order of the checking grid Y_{od}, and also to show that $v'(x) \leq 0$, $\forall\, x \in K$.*

   The second estimate (5.50):

   Substitute for the values $c, F, D_1, D_2, \boldsymbol{\beta}$ in the formula of the second derivative of the

orbital derivative (5.15)

$$\begin{aligned}\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| &\leq \boldsymbol{\beta}\Big(157870.623\ c^4F^2 + 19358.745\ c^3FD_1 + 1124.797\ c^2FD_2\Big),\\ &= 0.4105\ \Big(157870.623\ (4.717)^2 + 19358.745\ (4.717)\ (5.657) + 1124.797\\ &(4.717)\ (9.2195)\Big),\\ &\leq 1.7\times 10^6. \end{aligned} \tag{6.1}$$

Since d is even, the best distribution of grid points in Y_{od} would be the square configuration. Thus, to find h_1, we pass the value of (6.1) into the corresponding formula of the second estimate (5.50)

$$\begin{aligned} v'\left(\sum_{i=1}^{k}\lambda_i x_i\right) &\leq \max_{y\in Y_{od}} v'(y) + \left(\max_{z\in\mathcal{T}}\max_{r,s=1,2}\left|\frac{\partial^2 v'(z)}{\partial x_r \partial x_s}\right|\right)\ h_1^2,\\ &= \max_{y\in Y_{od}} v'(y) + \Big(1.7\times 10^6\Big)\ h_1^2 \end{aligned} \tag{6.2}$$

Assume that the orbital derivative of constructed function v' is a good approximation to $V' \Rightarrow \max\limits_{y\in Y_{od}} v'(y) \approx -\epsilon$, where $\epsilon > 0$ is a constant, thus

$$1.7\times 10^6\ h_1^2 \leq \epsilon\ \Rightarrow h_1 \leq \sqrt{\frac{\epsilon}{1.7}}\ 10^{-3}.$$

The estimated value of h_1 indicates the supposed density of the checking grid Y_{od} in which we use to calculate the maximum value of the orbital derivative. We have chosen the density of Y_{od} to be $h_1 = 0.417\ \left(10^{-4}\right)$, then checking the sign of the orbital derivative over Y_{od} gave $\max\limits_{y\in Y_{od}} v'(y) = -0.0038$. Therefore, (6.2) yields

$$\begin{aligned} v'(x) &\leq -0.0038 + \left(1.7\times 10^6\right)\ \left(0.417\times 10^{-4}\right)^2,\\ &= -0.0038 + 0.003 = -0.0008. \end{aligned}$$

Which shows that $v'(x) < 0$, for all $x \in K\backslash E_{nh}$, where $E_{nh} = [-0.1, 0.1]^2$ .Thus the constructed function v is a Lyapunov function.

Example 6.2. *Considering again Example 3.3.*

In this example, we have obtained an RBF approximant to the Lyapunov function satisfying $V'(x) = -1$ using the refinement algorithm. The refinement process started with an initial

grid $X_1 = 36$ grid points and terminated at $X_4 = 88$ points, after four refinement steps. The constructed function v has negative orbital derivative on a checking grid X_{check} with $h_{check} = 10^{-3}$. However, a rigorous checking will be carried out using the verification estimates.

1. *Recall the quantities $F = 3.57,\ D_1 = 4.98,\ D_2 = 5.69$.*
2. *We calculate the value of $\boldsymbol{\beta} = \sum_{k=1}^{88} |\beta_k| = 1.7372$.*
3. *Use the verification estimates to determine the order of the checking grid Y_{od}, and also to show if $v'(x) \leq 0,\ \forall\, x \in K$.*

The second estimate (5.50):

Calculate the second derivative of the orbital derivative

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq 1.7372 \times 2379065.026 = 4.1 \times 10^6.$$

Since d is even, the best distribution of grid points in Y_{od} would be the square configuration. Thus,

$$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \left(4.1 \times 10^6\right) h_1^2 \tag{6.3}$$

If v' is a good approximation to $V' \Rightarrow \max\limits_{y \in Y_{od}} v'(y) = -1$, then

$$h_1 \leq \sqrt{\frac{1}{4.1}}\, 10^{-3} \approx 0.5 \times 10^{-3}.$$

The supposed density of the checking grid Y_{od} in which we use to calculate the maximum value of the orbital derivative. We have chosen the density of Y_{od} to be $h_1 = 0.405\ \left(10^{-4}\right)$, then checking the sign of the orbital derivative over Y_{od} gave $\max\limits_{y \in Y_{od}} v'(y) = -0.0142$. Therefore, (6.3) yields

$$\begin{aligned} v'(x) &\leq -0.0142 + \left(4.1 \times 10^6\right)\ \left(0.405 \times 10^{-4}\right)^2, \\ &= -0.0142 + 0.0067 = -0.0075. \end{aligned}$$

Thus, $v'(x) < 0$, for all $x \in K \backslash [-0.1, 0.1]^2$, and the constructed function v is a Lyapunov function.

6.2.2 Examples solved with the modified algorithms

In Chapter 4, we have dealt with different cases of the unsuccessful termination of the refinement algorithm, i.e. where the refinement terminates without constructing a Lyapunov function. Fortunately, with the modified refinement algorithms we were able to construct RBF approximants that have negative orbital derivative on a checking grid X_{check} of size h_{check}. However, we need to verify the negativity of the orbital derivative of v on a more definite checking grid of density determined by using the verification estimates. In the following examples, we are going to apply the verification estimates on the functions v constructed with the first and second modified algorithms.

Example 6.3. *The case considered in Example 4.1 and Example 4.3 is where the refinement (for the system) terminated at 93 grid points with no Lyapunov function. The modified algorithms have successfully constructed RBF approximants v with a total of 104 points. Moreover, these approximants had negative orbital derivative on checking grid X_{check} with $h_{check} = 10^{-2}$. In this example, we will carry out a further negativity checking on a more accurate grid of a specified density h_1. Recall the quantities $F = 4.717$, $D_1 = 5.657$, $D_2 = 9.2195$, and $\boldsymbol{\beta} = \sum_{k=1}^{104} |\beta_k| = 0.5429$.*

1. ***The first modified algorithm.***

   The second estimate (5.50):

   - *The second derivative of the orbital derivative is*

   $$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq 0.5429 \times 4078121.1 = 2.2 \times 10^6. \tag{6.4}$$

   - *The estimate for the square configuration of grid points in Y_{od}*

   $$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \left(2.2 \times 10^6\right) h_1^2 \tag{6.5}$$

   - *From (6.5), the estimated order of h_1 is*

   $$h_1 \leq \sqrt{\frac{\epsilon}{2.2}}\, 10^{-3}.$$

   - *We have chosen $h_1 = 0.83\ \left(10^{-4}\right)$, then checking the sign of the orbital derivative*

over Y_{od} *gave* $\max_{y\in Y_{od}} v'(y) = -0.0274$. *Thus, (6.5) yields*

$$\begin{aligned} v'(x) &\leq -0.0274 + \left(2.2\times 10^{6}\right)\ \left(0.83\times 10^{-4}\right)^{2}, \\ &= -0.0274 + 0.0152 = -0.0122. \end{aligned}$$

Which shows that $v'(x) < 0$ *for all* $x \in K\backslash E_{nh}$, *where* $E_{nh} = [-0.1, 0.1]^2$. *Thus the constructed function* v *with the first modified algorithm is a Lyapunov function.*

2. ***The second modified algorithm.***

   The second estimate (5.50):

   - $\boldsymbol{\beta} = \sum_{k=1}^{104} |\beta_k| = 0.5459$.
   - *The second derivative of the orbital derivative is*

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq 0.5459 \times 4078121.1 = 2.2\times 10^{6}. \tag{6.6}$$

   - *The estimate for the square configuration of grid points in* Y_{od}

$$v'(x) \leq \max_{y\in Y_{od}} v'(y) + \left(2.2\times 10^{6}\right)\ h_1^2 \tag{6.7}$$

   - *From (6.7), the estimated order of* h_1 *is*

$$h_1 \leq \sqrt{\frac{\epsilon}{2.2}}\ 10^{-3}.$$

   - *we have set the density of the checking grid* Y_{od} *to* $h_1 = 0.83\ \left(10^{-4}\right)$. *Then, the checking process yields that* $\max_{y\in Y_{od}} v'(y) = -0.0269$. *As a result, (6.7) gives*

$$\begin{aligned} v'(x) &\leq -0.0269 + \left(2.2\times 10^{6}\right)\ \left(0.83\times 10^{-4}\right)^{2}, \\ &= -0.0269 + 0.0152 = -0.0117. \end{aligned}$$

   Which shows that $v'(x) < 0$, *for all* $x \in K\backslash E_{nh}$. *Thus the constructed function* v *with the second modified algorithm is a Lyapunov function.*

Example 6.4. *In Example 4.2 and Example 4.4, we have solved the case where the refinement stopped at 60 grid points without constructing a Lyapunov function with the first and second modified algorithm, respectively. Both algorithms have managed to construct*

RBF approximants v with a total of 96 grid points. Moreover, these approximants have negative orbital derivative on checking grid X_{check} with $h_{check} = 10^{-2}$. In this example, we verify the negativity of these RBf approximants on a different and more accurate checking grid with specified density h_1.

Recall the quantities $F = 3.57,\ D_1 = 4.98,\ D_2 = 5.69$, and $\boldsymbol{\beta} = \sum_{k=1}^{96} |\beta_k| = 1.7876$.

1. ***The first modified algorithm.***

   The second estimate (5.50):

   - *The second derivative of the orbital derivative is*

   $$\left| \frac{\partial^2 v'(x)}{\partial x_i \partial x_j} \right| \leq 1.7876 \times 2379065.026 = 4.3 \times 10^6. \tag{6.8}$$

   - *The estimate for the square configuration of grid points in Y_{od}*

   $$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \left(4.3 \times 10^6\right)\ h_1^2 \tag{6.9}$$

   - *From (6.9), the estimated order of h_1 is*

   $$h_1 \leq \sqrt{\frac{1}{4.3}}\ 10^{-3} \approx 0.5 \times 10^{-3}.$$

   - *We have chosen $h_1 = 0.9\ \left(10^{-4}\right)$, then checking the sign of the orbital derivative over Y_{od} gave $\max\limits_{y \in Y_{od}} v'(y) = -0.0808$. Thus, (6.9) yields*

   $$\begin{aligned} v'(x) &\leq -0.0808 + \left(4.3 \times 10^6\right)\ \left(0.9 \times 10^{-4}\right)^2, \\ &= -0.0808 + 0.035 = -0.0458. \end{aligned}$$

   Which shows that $v'(x) < 0$ for all $x \in K \backslash E_{nh}$, where $E_{nh} = [-0.1, 0.1]^2$. Thus the constructed function v with the first modified algorithm is a Lyapunov function.

2. ***The second modified algorithm.***

   The second estimate (5.50):

   - $\boldsymbol{\beta} = \sum_{k=1}^{96} |\beta_k| = 1.7123$.

- *The second derivative of the orbital derivative is*

$$\left|\frac{\partial^2 v'(x)}{\partial x_i \partial x_j}\right| \leq 1.7123 \times 2379065.026 = 4.1 \times 10^6. \tag{6.10}$$

- *The estimate for the square configuration of grid points in* Y_{od}

$$v'(x) \leq \max_{y \in Y_{od}} v'(y) + \left(4.1 \times 10^6\right) h_1^2 \tag{6.11}$$

- *From (6.11), the estimated order of* h_1 *is*

$$h_1 \leq \sqrt{\frac{1}{4.1}}\, 10^{-3} \approx 0.5 \times 10^{-3}.$$

- *we have set the density of the checking grid* Y_{od} *to* $h_1 = 0.747\ \left(10^{-4}\right)$*. Then, the checking process yields that* $\max_{y \in Y_{od}} v'(y) = -0.0484$*. As a result, (6.11) gives*

$$\begin{aligned} v'(x) &\leq -0.0484 + \left(4.1 \times 10^6\right)\ \left(0.747 \times 10^{-4}\right)^2, \\ &= -0.0484 + 0.023 = -0.0254. \end{aligned}$$

  Which shows that $v'(x) < 0$*, for all* $x \in K \backslash E_{nh}$*, thus the constructed function* v *with the second modified algorithm is a Lyapunov function*

So far, the verification estimates have shown efficient results regarding the functions constructed with the refinement and the modified algorithms and have always proved them to be Lyapunov functions. However, there is no guarantee that this combination method will always terminate successfully with a Lyapunov function.

Chapter 7

Conclusion

In this thesis we have introduced a refinement algorithm for the Radial Basis Function method to construct Lyapunov functions. The recursive refinement of the grid is done by considering the Voronoi vertices of the previous grid as possible new grid points, which are added if the orbital derivative of the previous approximation is non-negative. The algorithm terminates, if no points are added.

Compared to using a regular grid, the evaluation of the RBF approximant with the refinement algorithm is more efficient since it is able to reduce the required number of grid points by nearly a factor of 4 in some examples and a factor of 2 in others, depending on the system we are solving and which Lyapunov function we approximate (i.e., T' or Q'). This reduces the computational effort and time considerably when evaluating the constructed Lyapunov function. It even succeeded in constructing a Lyapunov function if the grid was placed in a set, which was not a subset of the domain of attraction.

Some starting grids may cause the refinement algorithm to terminate without constructing a Lyapunov function. Since all Voronoi vertices already have negative orbital derivative, we end up having some patches, in between the grid points, where the orbital derivative is positive. Thus, to overcome this problem we presented two modified refinement algorithms which make use of data clustering techniques to place grid points only in these patches as follows:
Consider each patch as a cluster, then determine the centre of the cluster by using either the k-means clustering or the subtractive clustering techniques. Add all the cluster centres to the previous grid points (obtained from the terminated refinement process) and

use this set to construct a new RBF approximant. If the constructed function still has areas of positive orbital derivative, implement the original refinement algorithm until it terminates. The refinement with both techniques (Voronoi vertices and cluster centres) continues until a Lyapunov function is found. The aim of these modified algorithms is to keep adding points to the grid, when the original refinement algorithm fails to do so. Numerical examples showed that the second modified algorithm, which uses the subtractive clustering, is much easier to implement and faster than the first one, and it is thus preferable when dealing with high-dimensional spaces.

A further improvement to this construction method was done by providing reliable verification estimates for the negativity of the orbital derivative of the constructed function with either a regular grid of points or with the refinement algorithms.
These estimates rely on the first and second derivatives of the orbital derivative and they provide an estimate on the density of the checking grid, where the sign and value of the orbital derivative of the constructed function is checked on finitely many points.
The effects of different factors such as different norms and different distributions of grid points have been analysed. We have obtained improved formulas of the verification estimates by using the combination of norms where $p = \infty \backslash \max$ and $q = 1$. Moreover, considering the value of the fill distance under the L_1-norm as well as the number of points in which different configurations require to fill in a hypercube $[-1, 1]^d \subset \mathbb{R}^d$, it turned out that the optimal distribution of grid points for the first estimate and the second estimate in odd dimensions is the body centred square configuration. While the square configuration is optimal for the second estimate in even dimensions.

Finally, we have proved that the verification estimates together with the RBF method to construct Lyapunov functions with a regular grid of points provide a successful technique to construct and verify a Lyapunov function for non-linear ODEs in $\mathbb{R}^d$ with exponentially stable equilibria.

This thesis has established the first refinement algorithm for the construction of Lyapunov functions using Radial Basis Functions. It is a considerable improvement from the regular grid, that was used until now, and provides a systematic way to reduce the number of grid points and tackle larger problems in higher dimensions. Moreover, the verification estimates combined with this construction method is of great advantage as it

proves that the constructed function is truly a Lyapunov function. Therefore, arbitrary compact subsets of the domain of attraction can be determined through sub-level sets of the constructed Lyapunov function. The proposed methods were efficiently applied to numerical examples in 2 and 3 dimensions.

The refinement algorithms as well as the verification estimates are independent of the choice of the Radial Basis Function used in the construction method. Therefore, future work includes trying different RBFs and compare the results to the ones obtained with the Wendland functions.

Appendix A

The product functions $\Psi_{i,k}$

We need to find the maximal value of the product functions $\Psi_{i,k}$ defined in Section 5.1 by using the second derivative test. We will calculate the product functions with respect to the Wendland basis function $\psi_{6,4}$ and the Gaussian radial basis function. Note that, we consider positive and real roots only.

A.1 The product functions w.r.t the Wendland function $\psi_{6,4}$

- $\phi_1(r) \coloneqq (1-cr)_+^6\ r^4$.

  The first derivative is:

$$\begin{aligned}\frac{d\phi_1}{dr} &= -6\ c\ (1-cr)^5\ r^4 + 4\ (1-cr)^6\ r^3, \\ &= (1-cr)^5\ r^3\ (4-10cr) = 0 \end{aligned} \tag{A.1}$$

  The solutions of (A.1) are:

  $r_1 = 0$, $r_2 = \frac{1}{c}$, and $r_3 = \frac{2}{5c}$.

  The second derivative is:

$$\frac{d^2\phi_1}{dr^2} = -10c\ (1-cr)^5\ r^3 - 5c\ (4-10cr)\ (1-cr)^4\ r^3 + 3r^2\ (1-cr)^5\ (4-10cr).$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  <u>When $r_1 = 0$:</u> $\quad \frac{d^2\phi_1}{dr^2}(r_1) = 0,$

  <u>When $r_2 = \frac{1}{c}$:</u> $\quad \frac{d^2\phi_1}{dr^2}(r_2) = 0,$

When $r_3 = \frac{2}{5c}$: $\quad \frac{d^2\phi_1}{dr^2}(r_3) = -0.082944\frac{1}{c^2} < 0,$

Thus, the function ϕ_1 attains its maximum value at $r_3 = \frac{2}{5c}$ and the maximum value is

$$\Phi_1 = \phi_1(r_3) = (1 - \frac{2}{5})^6 \, (\frac{2}{5c})^4 = 0.001194 \, \frac{1}{c^4}$$

- $\phi_2(r) \coloneqq (1 - cr)_+^7 (1 + 7cr)\, r^3$.

  The first derivative is:

$$\begin{aligned}
\frac{d\phi_2}{dr} &= -7c \, (1 - cr)^6 \, r^3 \, (1 + 7cr) + 3r^2 \, (1 - cr)^7 \, (1 + 7cr) + 7c \, (1 - cr)^7 \, r^3, \\
&= (1 - cr)^6 \, r^2 \big[-7cr - 49c^2r^2 + 3 + 21cr - 3cr - 21c^2r^2 + 7cr - 7c^2r^2 \big], \\
&= (1 - cr)^6 \, r^2 \, [-77c^2r^2 + 18cr + 3] = 0 \qquad \text{(A.2)}
\end{aligned}$$

  The solutions of (A.2) are:
  $r_1 = 0$, $r_2 = \frac{1}{c}$, $r_3 = \frac{-2\sqrt{78}+9}{77c} < 0$, and $r_4 = \frac{2\sqrt{78}+9}{77c}$.
  The second derivative is:

$$\begin{aligned}
\frac{d^2\phi_2}{dr^2} &= -6cr^2 \, (1 - cr)^5 \, (-77c^2r^2 + 18cr + 3) + 2r \, (1 - cr)^6 \, (-77c^2r^2 + 18cr + 3) \\
&\quad + r^2 \, (1 - cr)^6 (-154c^2r + 18c), \\
&= r \, (1 - cr)^5 \, [770c^3r^3 - 470c^2r^2 + 30cr + 6].
\end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  When $r_1 = 0$: $\quad \frac{d^2\phi_2}{dr^2}(r_1) = 0,$

  When $r_2 = \frac{1}{c}$: $\quad \frac{d^2\phi_2}{dr^2}(r_2) = 0,$

  When $r_4 = \frac{2\sqrt{78}+9}{77c}$: $\quad \frac{d^2\phi_2}{dr^2}(r_4) = -0.33061\frac{1}{c} < 0,$

  Thus, the function ϕ_2 attains its maximum value at $r_4 = \frac{2\sqrt{78}+9}{77c}$ and the maximum value is

$$\Phi_2 = \phi_2(r_4) = 0.007253 \, \frac{1}{c^3}.$$

- $\phi_3(r) \coloneqq (1-cr)_+^7(1+7cr)\ r^2$.

  The first derivative is:

$$\begin{aligned}\frac{d\phi_3}{dr} &= -7c\ (1-cr)^6\ r^2\ (1+7cr) + 2r\ (1-cr)^7\ (1+7cr) + 7c\ (1-cr)^7\ r^2,\\ &= (1-cr)^6\ r\big[-7cr - 49c^2r^2 + 2 + 14cr - 2cr - 7c^2r^2 + 7cr - 7c^2r^2\big],\\ &= (1-cr)^6\ r^2\ [-70c^2r^2 + 12cr + 2] = 0 \qquad \text{(A.3)}\end{aligned}$$

  The solutions of (A.3) are:

  $r_1 = 0$, $r_2 = \frac{1}{c}$, $r_3 = \frac{-2\sqrt{11}+3}{35c} < 0$, and $r_4 = \frac{2\sqrt{11}+3}{35c}$.

  The second derivative is:

$$\begin{aligned}\frac{d^2\phi_3}{dr^2} &= -6cr\ (1-cr)^5\ (-70c^2r^2 + 12cr + 2) + (1-cr)^6\ (-70c^2r^2 + 12cr + 2) + r\ (1-cr)^6\\ &\quad (-140c^2r + 12c),\\ &= (1-cr)^5\ [630c^3r^3 - 306c^2r^2 + 10cr + 2].\end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  When $r_1 = 0$: $\quad \frac{d^2\phi_3}{dr^2}(r_1) = 2$,

  When $r_2 = \frac{1}{c}$: $\quad \frac{d^2\phi_3}{dr^2}(r_2) = 0$,

  When $r_4 = \frac{2\sqrt{11}+3}{35c}$: $\quad \frac{d^2\phi_3}{dr^2}(r_4) = -1.05846 < 0$,

  Thus, the function ϕ_3 has a maximum value at $r_4 = \frac{2\sqrt{11}+3}{35c}$

$$\Phi_3 = \phi_3(r_4) = 0.0233\ \frac{1}{c^2}.$$

- $\phi_4(r) \coloneqq (1-cr)_+^8(1+8cr+21c^2r^2)\ r^2$.

  The first derivative is:

$$\begin{aligned}\frac{d\phi_4}{dr} &= -8c\ (1-cr)^7\ r^2\ (1+8cr+21c^2r^2) + 2r\ (1-cr)^8\ (1+8cr+21c^2r^2) + (1-cr)^8\ r^2\\ &\quad (8c + 42c^2r),\\ &= (1-cr)^7\ r\ \big[-8cr\ (1+8cr+21c^2r^2) + 2(1-cr)\ (1+8cr+21c^2r^2)\\ &\quad + r\ (1-cr)\ (8c+42c^2r)\big],\\ &= (1-cr)^7\ r\ 2\ [-126c^3r^3 - 2c^2r^2 + 7cr + 1] = 0 \qquad \text{(A.4)}\end{aligned}$$

Calculate the roots of the cubic equation $-126x^3 - 2x^2 + 7x + 1 = 0$ where we set $x = cr$, using the Discriminant approach:
The discriminant of the cubic equation is

$$\Delta = 18\ abcd - 4\ b^3d + b^2c^2 - 4\ ac^3 - 27\ a^2d^2.$$

where $a = -126, b = -2, c = 7, d = 1$, then

$$\Delta = 31752 + 32 + 196 + 172872 - 428652 = -223800 < 0,$$

Thus, the equation has one real root and two non-real complex conjugate roots.
The real root is given by the general formula

$$x = \frac{-1}{3a}\ \left(b + C + \frac{\Delta_0}{C}\right).$$

where:

$$\begin{aligned} \Delta_0 &= b^2 - 3ac = 2650, \\ \Delta_1 &= 2b^3 - 9abc + 27a^2d = 412760, \\ \Delta_1^2 - 4\Delta_0^3 &= -27a^2\Delta = 95932317600, \\ C &= \sqrt[3]{\frac{\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3}}{2}} = 71.21976. \end{aligned}$$

Thus

$$x = 0.281557 \Rightarrow r_3 = \frac{0.281557}{c}.$$

The solutions of (A.4) are:
$r_1 = 0$, $r_2 = \frac{1}{c}$, and $r_3 = 0.281557\frac{1}{c}$.
The second derivative is:

$$\begin{aligned} \frac{d^2\phi_4}{dr^2} &= -7cr\ (1 - cr)^6\ (-252c^3r^3 - 4c^2r^2 + 14cr + 2) + (1 - cr)^7 \\ &\quad (-252c^3r^3 - 4c^2r^2 + 14cr + 2) + r\ (1 - cr)^7\ (-756c^3r^2 - 8c^2r + 14c), \\ &= (1 - cr)^6\ [2772c^4r^4 - 968c^3r^3 - 138c^2r^2 + 12cr + 2]. \end{aligned}$$

To find the maximum points we substitute for the stationary points in the second derivative.

When $r_1 = 0$: $\quad \frac{d^2\phi_4}{dr^2}(r_1) = 2,$

When $r_2 = \frac{1}{c}$: $\quad \frac{d^2\phi_4}{dr^2}(r_2) = 0,$

When $r_3 = 0.281557\frac{1}{c}$: $\quad \frac{d^2\phi_4}{dr^2}(r_3) = -1.34034 < 0,$

Thus, the function ϕ_4 attains its maximum value at $r_3 = 0.281557$

$$\Phi_4 = \phi_4(r_3) = 0.027667\ \frac{1}{c^2}.$$

- $\phi_5(r) \coloneqq (1-cr)_+^8(1 + 8cr + 21c^2r^2)\ r.$

  The first derivative is:

$$\begin{aligned}\frac{d\phi_5}{dr} &= -8c\ (1-cr)^7\ r\ (1 + 8cr + 21c^2r^2) + (1-cr)^8\ (1 + 8cr + 21c^2r^2) + (1-cr)^8\ r \\ &\quad (8c + 42c^2r), \\ &= (1-cr)^7\ \big[-8cr\ (1 + 8cr + 21c^2r^2) + (1-cr)\ (1 + 8cr + 21c^2r^2) \\ &\quad + r\ (1-cr)\ (8c + 42c^2r)\big], \\ &= (1-cr)^7\ [-231c^3r^3 - 17c^2r^2 + 7cr + 1] = 0 \end{aligned} \tag{A.5}$$

  Calculate the roots of the cubic equation $-231x^3 - 17x^2 + 7x + 1 = 0$ where we set $x = cr$, using the Discriminant approach:

  The discriminant of the cubic equation is

$$\Delta = 18\ abcd - 4\ b^3d + b^2c^2 - 4\ ac^3 - 27\ a^2d^2.$$

  where $a = -231, b = -17, c = 7, d = 1$, then

$$\Delta = 494802 + 19652 + 14161 + 316932 - 1440747 = -595200 < 0,$$

  Thus, the equation has one real root and two non-real complex conjugate roots.

  The real root is given by the general formula

$$x = \frac{-1}{3a}\ \left(b + C + \frac{\Delta_0}{C}\right).$$

where:

$$\begin{aligned}
\Delta_0 &= b^2 - 3ac = 5140, \\
\Delta_1 &= 2b^3 - 9abc + 27a^2d = 1183520, \\
\Delta_1^2 - 4\Delta_0^3 &= -27a^2\Delta = 857535614400, \\
C &= \sqrt[3]{\frac{\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3}}{2}} = 101.79348.
\end{aligned}$$

Thus

$$x = 0.195221 \Rightarrow r_2 = \frac{0.195221}{c}.$$

The solutions of (A.5) are:

$r_1 = \frac{1}{c}$, and $r_2 = 0.195221\frac{1}{c}$.

The second derivative is:

$$\begin{aligned}
\frac{d^2\phi_5}{dr^2} &= -7c\,(1-cr)^6\,(-231c^3r^3 - 17c^2r^2 + 7cr + 1) + (1-cr)^7 \quad (-693c^3r^2 - 34c^2r + 7c), \\
&= (1-cr)^6\,[2310c^4r^3 - 540c^3r^2 - 90c^2r].
\end{aligned}$$

To find the maximum points we substitute for the stationary points in the second derivative.

When $r_1 = \frac{1}{c}$: $\quad \frac{d^2\phi_5}{dr^2}(r_1) = 0$,

When $r_2 = 0.195221\frac{1}{c}$: $\quad \frac{d^2\phi_5}{dr^2}(r_2) = -5.69531\ c < 0$,

Thus, the function ϕ_5 attains its maximum value at $r_2 = 0.195221\frac{1}{c}$

$$\Phi_5 = \phi_5(r_2) = 0.115492\frac{1}{c}.$$

- $\phi_6(r) \coloneqq (1-cr)_+^8(1 + 8cr + 21c^2r^2)$.

  The first derivative is:

$$\begin{aligned}
\frac{d\phi_6}{dr} &= -8c\,(1-cr)^7\,(1 + 8cr + 21c^2r^2) + (1-cr)^8\,(8c + 42c^2r), \\
&= (1-cr)^7\,\big[-8c\,(1 + 8cr + 21c^2r^2) + (1-cr)\,(8c + 42c^2r)\big], \\
&= (1-cr)^7\,[-210c^3r^2 - 30c^2r] = 0 \qquad \text{(A.6)}
\end{aligned}$$

The solutions of (A.6) are:

$r_1 = 0$, $r_2 = \frac{1}{c}$, and $r_3 = \frac{-1}{7c} < 0$.

The second derivative is:

$$\begin{aligned}\frac{d^2\phi_6}{dr^2} &= -7c\,(1-cr)^6\,(-210c^3r^2 - 30c^2r) + (1-cr)^7\,(-420c^3r - 30c^2),\\ &= (1-cr)^6\,[1890c^4r^2 - 180c^3r - 30c^2].\end{aligned}$$

To find the maximum points we substitute for the stationary points in the second derivative.

When $r_1 = 0$: $\quad \frac{d^2\phi_6}{dr^2}(r_1) = -30c^2 < 0$,

When $r_2 = \frac{1}{c}$: $\quad \frac{d^2\phi_6}{dr^2}(r_2) = 0$,

Thus, the function ϕ_6 attains its maximum value at $r_1 = 0$

$$\Phi_6 = \phi_6(r_1) = 1.$$

- $\phi_7(r) \coloneqq (1-cr)_+^9(5 + 45cr + 159c^2r^2 + 231c^3r^3)$.

  The first derivative is:

$$\begin{aligned}\frac{d\phi_7}{dr} &= -9c\,(1-cr)^8\,(231c^3r^3 + 159c^2r^2 + 45cr + 5) + (1-cr)^9\,(693c^3r^2 + 318c^2r + 45c),\\ &= (1-cr)^8\,\big[-2079c^4r^3 - 1431c^3r^2 - 405c^2r - 45c + 693c^3r^2 + 318c^2r + 45c - 693c^4r^3\\ &\quad - 318c^3r^2 - 45c^2r\big],\\ &= (1-cr)^8\big(-132c^2r(21c^2r^2 + 8cr + 1)\big) = 0 \qquad \text{(A.7)}\end{aligned}$$

  The solutions of (A.7) are :

  $r_1 = 0$, $r_2 = \frac{1}{c}$, $r_3 = \frac{-4+\sqrt{5i}}{21c} \notin \mathbb{R}$, and $r_4 = \frac{-4-\sqrt{5i}}{21c} \notin \mathbb{R}$.

  The second derivative is:

$$\begin{aligned}\frac{d^2\phi_7}{dr^2} &= -8c\,(1-cr)^7\,(-2772c^4r^3 - 1056c^3r^2 - 132c^2r) + (1-cr)^8\\ &\quad (-8316c^4r^2 - 2112c^3r - 132c^2),\\ &= (1-cr)^7\,[30492c^5r^3 + 2244c^4r^2 - 924c^3r - 132c^2].\end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

When $r_1 = 0$: $\frac{d^2\phi_7}{dr^2}(r_1) = -132c^2 < 0$,

When $r_2 = \frac{1}{c}$: $\frac{d^2\phi_7}{dr^2}(r_2) = 0$,

Thus, the function ϕ_7 attains its maximum value at $r_1 = 0$,

$$\Phi_7 = \phi_7(r_1) = 5.$$

A.2 The product functions w.r.t the Gaussian function

- $\phi_1(r) \coloneqq e^{-\varepsilon^2\, r^2}\, r^4$.

  The first derivative is :

$$\begin{aligned}\frac{d\phi_1}{dr} &= -2\,\varepsilon^2\, r^5\, e^{-\varepsilon^2\, r^2} + 4\, r^3\, e^{-\varepsilon^2\, r^2},\\ &= e^{-\varepsilon^2\, r^2}\,[-2\varepsilon^2\, r^5 + 4\, r^3] = 0. \end{aligned} \tag{A.8}$$

  The solutions of (A.8) are:

  $r_1 = 0$, $r_2 = \sqrt{2}\,\frac{1}{\varepsilon}$, and $r_3 = -\sqrt{2}\,\frac{1}{\varepsilon} < 0$.

  The second derivative is:

$$\begin{aligned}\frac{d^2\phi_1}{dr^2} &= -2\,\varepsilon^2\, r\, e^{-\varepsilon^2\, r^2}(-2\,\varepsilon^2\, r^5 + 4\, r^3) + e^{-\varepsilon^2\, r^2}(-10\varepsilon^2\, r^4 + 12r^2),\\ &= e^{-\varepsilon^2\, r^2}(-18\,\varepsilon^2\, r^4 + 4\,\varepsilon^4\, r^6 + 12\, r^2).\end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  When $r_1 = 0$: $\frac{d^2\phi_1}{dr^2} = 0$,

  When $r_2 = \sqrt{2}\frac{1}{\varepsilon}$: $\frac{d^2\phi_1}{dr^2} = -2.16536\,\frac{1}{\varepsilon^2} < 0$,

  Thus the function ϕ_1 has a maximum value at $r_2 = \sqrt{2}\,\frac{1}{\varepsilon}$

$$\Phi_1 = \phi_1(r_2) = \phi_1(r_3) = e^{-2}\frac{4}{\varepsilon^4} = 0.5412\,\frac{1}{\varepsilon^4}.$$

- $\phi_2(r) \coloneqq e^{-\varepsilon^2\, r^2}\, r^3$.

  The first derivative is :

$$\begin{aligned}\frac{d\phi_2}{dr} &= -2\,\varepsilon^2\, r^4\, e^{-\varepsilon^2\, r^2} + 3\, r^2\, e^{-\varepsilon^2\, r^2},\\ &= e^{-\varepsilon^2\, r^2}\,[-2\varepsilon^2\, r^4 + 3\, r^2] = 0. \end{aligned} \tag{A.9}$$

The solutions of (A.9) are:

$r_1 = 0$, $r_2 = \sqrt{\frac{3}{2}}\ \frac{1}{\varepsilon}$, and $r_3 = -\sqrt{\frac{3}{2}}\ \frac{1}{\varepsilon} < 0$.

The second derivative is:

$$\begin{aligned}\frac{d^2\phi_2}{dr^2} &= -2\ \varepsilon^2\ r\ e^{-\varepsilon^2\ r^2}(-2\ \varepsilon^2\ r^4 + 3\ r^2) + e^{-\varepsilon^2\ r^2}(-8\varepsilon^2\ r^3 + 6r),\\ &= e^{-\varepsilon^2\ r^2}(4\ \varepsilon^4\ r^5 - 14\ \varepsilon^2\ r^3 + 6\ r).\end{aligned}$$

To find the maximum points we substitute for the stationary points in the second derivative.

When $r_1 = 0$: $\quad \frac{d^2\phi_2}{dr^2} = 0$,

When $r_2 = \sqrt{\frac{3}{2}}\ \frac{1}{\varepsilon}$: $\quad \frac{d^2\phi_2}{dr^2} = -1.63967\ \frac{1}{\varepsilon} < 0$,

Thus the function ϕ_2 attains its maximum value at $r_2 = \sqrt{\frac{3}{2}}\ \frac{1}{\varepsilon}$,

$$\Phi_2 = \phi_2(r_2) = e^{-\frac{3}{2}}\left(\frac{3}{2}\right)^{\frac{3}{2}}\ \frac{1}{\varepsilon^3} = 0.4099\ \frac{1}{\varepsilon^3}.$$

- $\phi_3(r) \coloneqq e^{-\varepsilon^2\ r^2}\ r^2$.

  The first derivative is :

$$\begin{aligned}\frac{d\phi_3}{dr} &= -2\ \varepsilon^2\ r^3\ e^{-\varepsilon^2\ r^2} + 2\ r\ e^{-\varepsilon^2\ r^2},\\ &= e^{-\varepsilon^2\ r^2}\ [-2\varepsilon^2\ r^3 + 2\ r] = 0. \qquad \text{(A.10)}\end{aligned}$$

  The solutions of (A.10) are:

  $r_1 = 0$, $r_2 = \frac{1}{\varepsilon}$, and $r_3 = -\frac{1}{\varepsilon} < 0$.

  The second derivative is:

$$\begin{aligned}\frac{d^2\phi_3}{dr^2} &= -2\ \varepsilon^2\ r\ e^{-\varepsilon^2\ r^2}(-2\ \varepsilon^2\ r^3 + 2\ r) + e^{-\varepsilon^2\ r^2}(-6\varepsilon^2\ r^2 + 2),\\ &= e^{-\varepsilon^2\ r^2}(4\ \varepsilon^4\ r^4 - 10\ \varepsilon^2\ r^2 + 2).\end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  When $r_1 = 0$: $\quad \frac{d^2\phi_3}{dr^2} = 2$,

When $r_2 = \frac{1}{\varepsilon}$: $\quad \frac{d^2\phi_3}{dr^2} = -1.47152 < 0,$

Thus the function ϕ_3 has a maximum value at $r_2 = \frac{1}{\varepsilon}$ and $r_3 = -\frac{1}{\varepsilon}$,

$$\Phi_3 = \phi_3(r_2) = \phi_3(r_3) = e^{-1}\ \frac{1}{\varepsilon^2} = 0.3679\ \frac{1}{\varepsilon^2}.$$

- $\phi_4(r) \coloneqq e^{-\varepsilon^2\ r^2}\ r.$

  The first derivative is :

$$\begin{aligned} \frac{d\phi_4}{dr} &= -2\ \varepsilon^2\ r^2\ e^{-\varepsilon^2\ r^2} + e^{-\varepsilon^2\ r^2}, \\ &= e^{-\varepsilon^2\ r^2}\ [-2\varepsilon^2\ r^2 + 1] = 0. \end{aligned} \tag{A.11}$$

  The solutions of (A.11) are:

  $r_1 = \frac{1}{\sqrt{2}}\ \frac{1}{\varepsilon}$, and $r_2 = -\frac{1}{\sqrt{2}}\ \frac{1}{\varepsilon} < 0.$

  The second derivative is:

$$\begin{aligned} \frac{d^2\phi_4}{dr^2} &= -2\ \varepsilon^2\ r\ e^{-\varepsilon^2\ r^2}(-2\ \varepsilon^2\ r^2 + 1) + e^{-\varepsilon^2\ r^2}(-4\varepsilon^2\ r), \\ &= e^{-\varepsilon^2\ r^2}(4\ \varepsilon^4\ r^3 - 6\ \varepsilon^2\ r). \end{aligned}$$

  To find the maximum points we substitute for the stationary points in the second derivative.

  When $r_1 = \frac{1}{\sqrt{2}}\ \frac{1}{\varepsilon}$: $\quad \frac{d^2\phi_4}{dr^2} = -1.71553\ \frac{1}{\varepsilon} < 0,$

  Thus the function ϕ_4 attains its maximum value at $r_1 = \frac{1}{\sqrt{2}}\ \frac{1}{\varepsilon}$,

$$\Phi_4 = \phi_4(r_1) = e^{-\frac{1}{2}}\frac{1}{\sqrt{2}}\ \frac{1}{\varepsilon} = 0.4289\ \frac{1}{\varepsilon}.$$

- $\phi_5(r) \coloneqq e^{-\varepsilon^2\ r^2}.$

  The first derivative is :

$$\frac{d\phi_5}{dr} = -2\ \varepsilon^2\ r\ e^{-\varepsilon^2\ r^2}. \tag{A.12}$$

The solution of (A.12) is:

$r_1 = 0$.

The second derivative is:

$$\begin{aligned}\frac{d^2\phi_5}{dr^2} &= -2\,\varepsilon^2\, e^{-\varepsilon^2\, r^2} - 2\,\varepsilon^2\, r\, (-2\,\varepsilon^2\, r)\, e^{-\varepsilon^2\, r^2},\\ &= e^{-\varepsilon^2\, r^2}(-2\varepsilon^2 + 4\,\varepsilon^4\, r^2).\end{aligned}$$

To find the maximum points we substitute for the stationary point in the second derivative.

When $r_1 = 0$: $\quad \frac{d^2\phi_5}{dr^2} = -2 < 0.$

Thus the function ϕ_5 attains its maximum value at $r_1 = 0$,

$$\Phi_5 = \phi_5(r_1) = 1.$$

Appendix B

The quantities F, D_1, and D_2

In this appendix we give the detailed calculations of the quantities F, D_1, and D_2, involved in the formulas of the first and second derivatives of the orbital derivative.

Example B.1. *For the 2-dimensional system*

$$\begin{cases} \dot{x} = -x - 2y + x^3 \ \ = f_1, \\ \dot{y} = -y + \frac{1}{2}x^2 y + x^3 = f_2. \end{cases}$$

and the compact set $K = \{(x, y) \in \mathbb{R}^2 \ : \ |x| \leq 1 \ , \ |y| \leq 1\}$.

1. $F := \max\limits_{x \in K} \|f(x)\|_2$.

$$\begin{aligned} \|f(x)\|_2^2 = f_1^2 + f_2^2 &= (-x - 2y + x^3)^2 + (-y + \frac{1}{2}x^2 y + x^3)^2, \\ &= x^2 + 2xy - x^4 + 2xy + 4y^2 - 2yx^3 - x^4 - 2x^3 y + x^6 + y^2 - \frac{1}{2}x^2 y^2 - x^3 y \\ &- \frac{1}{2}x^2 y^2 + \frac{1}{4}x^4 y^2 + \frac{1}{2}x^5 y - x^3 y + \frac{1}{2}x^5 y + x^6, \\ &= \mid x^2 + 4xy - 2x^4 + 5y^2 - 6x^3 y + 2x^6 - x^2 y^2 + \frac{1}{4}x^4 y^2 + x^5 y \mid, \\ &\leq |x|^2 + 4|x| \ |y| + 2|x|^4 + 5|y|^2 + 6|x|^3 \ |y| + 2|x|^6 + |x|^2 \ |y|^2 \\ &+ \frac{1}{4}|x|^4 \ |y|^2 + |x|^5 \ |y|, \\ &\leq 1 + 4 + 2 + 5 + 6 + 2 + 1 + \frac{1}{4} + 1 = 22.25, \end{aligned}$$

$F := \max\limits_{x \in K} \|f(x)\|_2 \leq \sqrt{22.25} = 4.717.$

2. $D_1 := \max\limits_{(x,y) \in K} \max\limits_{j=1,2} \|\frac{\partial f(x)}{\partial x_j}\|_2 = \max\limits_{(x,y) \in K} \left(\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2}, \sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2} \right)$.

$\frac{\partial f_1}{\partial x} = -1 + 3x^2, \;\; \frac{\partial f_2}{\partial x} = xy + 3x^2, \;\; \frac{\partial f_1}{\partial y} = -2, \;\; \frac{\partial f_2}{\partial y} = -1 + \frac{1}{2}x^2.$

$$\begin{aligned}(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2 &= (-1 + 3x^2)^2 + (xy + 3x^2)^2, \\ &= 1 - 3x^2 - 3x^2 + 9x^4 + x^2y^2 + 3x^3y + 3x^3y + 9x^4, \\ &= |\ 1 - 6x^2 + 18x^4 + x^2y^2 + 6x^3y\ |, \\ &\leq 1 + 6|x|^2 + 18|x|^4 + |x|^2\ |y|^2 + 6|x|^3\ |y| \leq 1 + 6 + 18 + 1 + 6 = 32,\end{aligned}$$

$$\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2} \leq 5.657.$$

$$\begin{aligned}(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2 &= (-2)^2 + (-1 + \frac{1}{2}x^2)^2, \\ &= 4 + 1 - x^2 + \frac{1}{4}x^4 = |\ 5 - x^2 + \frac{1}{4}x^4\ |, \\ &\leq 5 + |x|^2 + \frac{1}{4}|x|^4 \leq 5 + 1 + \frac{1}{4} = 6.25,\end{aligned}$$

$$\sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2} \leq 2.5.$$

$$D_1 = \max_{(x,y)\in K}\Big(5.657\ ,\ 2.5\Big) = 5.657.$$

3. $D_2 := \max\limits_{(x,y)\in K}\max\limits_{i,j=1,2} \|\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\|_2 = \max\limits_{(x,y)\in K}\Big(\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2},$
$\sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2}\Big).$

$\frac{\partial^2 f_1}{\partial x^2} = 6x, \;\; \frac{\partial^2 f_1}{\partial x \partial y} = 0, \;\; \frac{\partial^2 f_1}{\partial y \partial x} = 0, \;\; \frac{\partial^2 f_1}{\partial y^2} = 0.$
$\frac{\partial^2 f_2}{\partial x^2} = y + 6x, \;\; \frac{\partial^2 f_2}{\partial x \partial y} = x, \;\; \frac{\partial^2 f_2}{\partial y \partial x} = x, \;\; \frac{\partial^2 f_2}{\partial y^2} = 0.$

$$\begin{aligned}(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2 &= (6x)^2 + (y + 6x)^2, \\ &= 36x^2 + y^2 + 12xy + 36x^2 = |\ 72x^2 + 12xy + y^2\ |, \\ &\leq 72|x|^2 + 12|x|\ |y| + |y|^2 \leq 72 + 12 + 1 = 85,\end{aligned}$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2} \leq 9.2195.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2} = \sqrt{x^2} = \sqrt{|x|^2} \leq 1.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2} = \sqrt{x^2} = \sqrt{|x|^2} \leq 1.$$

$$\sqrt{(\frac{\partial^2 f_2}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2} = 0.$$

$$D_2 = \max_{(x,y)\in K} \Big(9.2195,\ 1,\ 0\Big) = 9.2195.$$

Example B.2. *For the 2-dimensional system*

$$\begin{cases} \dot{x} = -x(1 - x^2 - y^2) - y = f_1, \\ \dot{y} = -y(1 - x^2 - y^2) + x = f_2. \end{cases}$$

and the compact set $K = \{(x, y) \in \mathbb{R}^2 \ : \ |x| \leq 0.9 \ , \ |y| \leq 0.9\}$.

1. $F := \max_{x\in K} \|f(x)\|_2$.

$$\begin{aligned} \|f(x)\|_2^2 = f_1^2 + f_2^2 &= (-x(1 - x^2 - y^2) - y)^2 + (-y(1 - x^2 - y^2) + x)^2, \\ &= x^2 - x^4 - x^2y^2 + xy - x^4 + x^6 + x^4y^2 - x^3y - x^2y^2 + x^4y^2 - xy^3 \\ &+ xy - x^3y - xy^3 + y^2 + y^2 - y^2x^2 - y^4 - xy - x^2y^2 + y^2x^4 + y^4x^2 \\ &+ yx^3 - y^4 + y^4x^2 + y^6 + xy^3 - xy + x^3y + y^3x + x^2, \\ &= |\ 2x^2 - 2x^4 - 4x^2y^2 + x^6 + 3x^4y^2 + 3x^2y^4 + 2y^2 - 2y^4 + y^6\ |, \\ &\leq 2|x|^2 + 2|x|^4 + 4|x|^2\ |y|^2 + |x|^6 + 3|x|^4\ |y|^2 + 3|x|^2\ |y|^4 + 2|y|^2 + 2|y|^4 \\ &+ |y|^6, \\ &\leq 2(0.9)^2 + 2(0.9)^4 + 4(0.9)^4 + (0.9)^6 + 3(0.9)^6 + 3(0.9)^6 + 2(0.9)^2 \\ &+ 2(0.9)^4 + (0.9)^6 = 12.740. \end{aligned}$$

$F := \max_{x\in K} \|f(x)\|_2 \leq \sqrt{12.740} = 3.57.$

2. $D_1 := \max_{(x,y)\in K} \max_{j=1,2} \|\frac{\partial f(x)}{\partial x_j}\|_2 = \max_{(x,y)\in K} \Big(\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2}, \sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2}\Big)$.

$\frac{\partial f_1}{\partial x} = -1 + 3x^2 + y^2, \;\; \frac{\partial f_2}{\partial x} = 2xy + 1, \;\; \frac{\partial f_1}{\partial y} = 2xy - 1, \;\; \frac{\partial f_2}{\partial y} = -1 + x^2 + 3y^2.$

$$\begin{aligned}
(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2 &= (-1 + 3x^2 + y^2)^2 + (2xy + 1)^2, \\
&= \mid 2 - 6x^2 - 2y^2 + 9x^4 + 10x^2y^2 + 4xy + y^4 \mid, \\
&\leq 2 + 6|x|^2 + 2|y|^2 + 9|x|^4 + 10|x|^2 \, |y|^2 + 4|x| \, |y| + |y|^4, \\
&\leq 2 + 6(0.9)^2 + 2(0.9)^2 + 9(0.9)^4 + 10(0.9)^2(0.9)^2 + 4|(0.9)^2 + (0.9)^4 \\
&= 24.8424, \\
\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2} &\leq 4.984.
\end{aligned}$$

$$\begin{aligned}
(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2 &= (-1 + x^2 + 3y^2)^2 + (2xy - 1)^2, \\
&= \mid 2 - 2x^2 - 6y^2 + x^4 + 10x^2y^2 + 9y^4 - 4xy \mid, \\
&\leq 2 + 2|x|^2 + 6|y|^2 + |x|^4 + 10|x|^2 \, |y|^2 + 9|y|^4 + 4|x| \, |y|, \\
&leq 2 + 2(0.9)^2 + 6(0.9)^2 + (0.9)^4 + 10|(0.9)^4 + 9(0.9)^4 + 4(0.9)^2 = 24.8424, \\
\sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2} &\leq 4.984.
\end{aligned}$$

$D_1 = \max\limits_{(x,y)\in K} \Big(4.984, \; 4.984\Big) = 4.984.$

3. $D_2 := \max\limits_{(x,y)\in K} \max\limits_{i,j=1,2} \| \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \|_2 = \max\limits_{(x,y)\in K} \Big(\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2} \Big).$

$\frac{\partial^2 f_1}{\partial x^2} = 6x, \;\; \frac{\partial^2 f_1}{\partial x \partial y} = 2y, \;\; \frac{\partial^2 f_1}{\partial y \partial x} = 2y, \;\; \frac{\partial^2 f_1}{\partial y^2} = 2x.$
$\frac{\partial^2 f_2}{\partial x^2} = 2y, \;\; \frac{\partial^2 f_2}{\partial x \partial y} = 2x, \;\; \frac{\partial^2 f_2}{\partial y \partial x} = 2x, \;\; \frac{\partial^2 f_2}{\partial y^2} = 6y.$

$$
\begin{aligned}
(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2 &= (6x)^2 + (2y)^2 = \mid 36x^2 + 4y^2 \mid, \\
&\leq 36|x|^2 + 4|y|^2 \leq 36(0.9)^2 + 4(0.9)^2 = 32.4, \\
\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2} &\leq 5.692. \\
(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2 &= (2y)^2 + (2x)^2 = \mid 4y^2 + 4x^2 \mid, \\
&\leq 4|y|^2 + 4|x|^2 \leq 4(0.9)^2 + 4(0.9)^2 = 6.48, \\
\sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2} &\leq 2.546. \\
\sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2} &\leq 2.546. \\
\sqrt{(\frac{\partial^2 f_1}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2} &\leq 5.692.
\end{aligned}
$$

$$
D_2 = \max_{(x,y)\in K} \Big(5.692,\ 2.546\Big) = 5.692.
$$

Example B.3. *For the 3-dimensional system*

$$
\begin{cases}
\dot{x} = x(x^2 + y^2 - 1) - y(z^2 + 1) = f_1, \\
\dot{y} = y(x^2 + y^2 - 1) + x(z^2 + 1) = f_2, \\
\dot{z} = 10z(z^2 - 1) = f_3.
\end{cases}
$$

and the compact set $K = \{(x, y, z) \in \mathbb{R}^3 \ :\ |x| \leq 0.9,\ |y| \leq 0.9,\ |z| \leq 0.9\}$.

1. $F := \max_{x \in K} \|f(x)\|_2$.

$$\begin{aligned}
\|f(x)\|_2^2 = f_1^2 + f_2^2 + f_3^2 &= (x(x^2+y^2-1) - y(z^2+1))^2 + (y(x^2+y^2-1) \\
&+ x(z^2+1))^2 + (10z(z^2-1))^2, \\
&= |\ x^6 + 3x^4y^2 - 2x^4 + 2x^2y^4 - 4x^2y^2 - xy^2z^2 + 2x^2 + xy^3z^2 + 2y^2z^4 \\
&+ 2y^2z^2 + y^6 + 2y^2 - 2y^4 + 2x^2z^2 + 100z^6 - 200z^4 + 100z^2\ |, \\
&\leq |x|^6 + 3|x|^4\ |y|^2 + 2|x|^4 + 2|x|^2\ |y|^4 + 4|x|^2\ |y|^2 + |x|\ |y|^2\ |z|^2 + 2|x|^2 \\
&+ |x|\ |y|^3\ |z|^2 + 2|y|^2\ |z|^4 + 2|y|^2\ |z|^2 + |y|^6 + 2|y|^2 + 2|y|^4 + 2|x|^2\ |z|^2 \\
&+ 100|z|^6 + 200|z|^4 + 100|z|^2, \\
&\leq (0.9)^6 + 3(0.9)^6 + 2(0.9)^4 + 2(0.9)^6 + 4(0.9)^4 + (0.9)^5 + 2(0.9)^2 + (0.9)^6 \\
&+ 2(0.9)^6 + 2(0.9)^4 + (0.9)^6 + 2(0.9)^2 + 2(0.9)^4 + 2(0.9)^4 + 100(0.9)^6 \\
&+ 200(0.9)^4 + 100(0.9)^2 = 282.3803,
\end{aligned}$$

$F := \max_{x \in K} \|f(x)\|_2 \leq \sqrt{282.3803} = 16.804.$

2. $D_1 := \max_{(x,y,z) \in K} \max_{j=1,2,3} \|\frac{\partial f(x)}{\partial x_j}\|_2 = \max_{(x,y,z) \in K} \Big(\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2 + (\frac{\partial f_3}{\partial x})^2}, \sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2 + (\frac{\partial f_3}{\partial y})^2}, \sqrt{(\frac{\partial f_1}{\partial z})^2 + (\frac{\partial f_2}{\partial z})^2 + (\frac{\partial f_3}{\partial z})^2} \Big)$.

$$\begin{aligned}
&\tfrac{\partial f_1}{\partial x} = 3x^2 + y^2 - 1, \quad \tfrac{\partial f_2}{\partial x} = 2xy + z^2 + 1, \quad \tfrac{\partial f_3}{\partial x} = 0, \\
&\tfrac{\partial f_1}{\partial y} = 2xy - z^2 - 1, \quad \tfrac{\partial f_2}{\partial y} = x^2 + 3y^2 - 1, \quad \tfrac{\partial f_3}{\partial y} = 0, \\
&\tfrac{\partial f_1}{\partial z} = -2yz, \quad \tfrac{\partial f_2}{\partial z} = 2xz, \quad \tfrac{\partial f_3}{\partial z} = 30z^2 - 10.
\end{aligned}$$

$$
\begin{aligned}
(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2 + (\frac{\partial f_3}{\partial x})^2 &= (3x^2 + y^2 - 1)^2 + (2xy + z^2 + 1)^2, \\
&= |\ 9x^4 + 10x^2y^2 - 6x^2 + y^4 - 2y^2 + 4xyz^2 + 4xy + z^4 + 2z^2 + 2\ |, \\
&\leq 9|x|^4 + 10|x|^2|y|^2 + 6|x|^2 + |y|^4 + 2|y|^2 + 4|x|\ |y|\ |z|^2 \\
&+ 4|x|\ |y| + |z|^4 + 2|z|^2 + 2, \\
&\leq 9(0.9)^4 + 10(0.9)^4 + 6(0.9)^2 + (0.9)^4 + 2(0.9)^2 + 4(0.9)^4 \\
&+ 4(0.9)^2 + (0.9)^4 + 2(0.9)^2 + 2 = 25.6925, \\
\sqrt{(\frac{\partial f_1}{\partial x})^2 + (\frac{\partial f_2}{\partial x})^2 + (\frac{\partial f_3}{\partial x})^2} &\leq 5.069
\end{aligned}
$$

$$
\begin{aligned}
(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2 + (\frac{\partial f_3}{\partial y})^2 &= (2xy - z^2 - 1)^2 + (x^2 + 3y^2 - 1)^2, \\
&= |\ 10x^2y^2 - 4xyz^2 - 4xy + z^4 + 2z^2 + x^4 - 2x^2 + 9y^4 - 6y^2 + 2\ |, \\
&\leq 10|x|^2|y|^2 + 4|x|\ |y|\ |z|^2 + 4|x|\ |y| + |z|^4 + 2|z|^2 + |x|^4 + 2|x|^2 \\
&+ 9|y|^4 + 6|y|^2 + 2, \\
&\leq 10(0.9)^4 + 4(0.9)^4 + 4(0.9)^2 + (0.9)^4 + 2(0.9)^2 + (0.9)^4 + 2(0.9)^2 \\
&+ 9(0.9)^4 + 6(0.9)^2 + 2 = 25.6925, \\
\sqrt{(\frac{\partial f_1}{\partial y})^2 + (\frac{\partial f_2}{\partial y})^2 + (\frac{\partial f_3}{\partial y})^2} &\leq 5.069.
\end{aligned}
$$

$$
\begin{aligned}
(\frac{\partial f_1}{\partial z})^2 + (\frac{\partial f_2}{\partial z})^2 + (\frac{\partial f_3}{\partial z})^2 &= (-2yz)^2 + (2xz)^2 + (30z^2 - 10)^2, \\
&= |\ 4y^2z^2 + 4x^2z^2 + 900z^4 - 600z^2 + 100\ |, \\
&\leq 4|y|^2|z|^2 + 4|x|^2|z|^2 + 900|z|^4 + 600|z|^2 + 100, \\
&\leq 4(0.9)^4 + 4(0.9)^4 + 900(0.9)^4 + 600(0.9)^2 + 100 = 1181.7388, \\
\sqrt{(\frac{\partial f_1}{\partial z})^2 + (\frac{\partial f_2}{\partial z})^2 + (\frac{\partial f_3}{\partial z})^2} &\leq 34.376
\end{aligned}
$$

$$D_1 = \max_{(x,y,z)\in K} \Big(5.069,\ 34.376\Big) = 34.376.$$

3. $D_2 := \max\limits_{(x,y)\in K} \max\limits_{i,j=1,2} \|\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\|_2 = \max\limits_{(x,y)\in K} \Big(\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2 + (\frac{\partial^2 f_3}{\partial x^2})^2},$
$\sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2 + (\frac{\partial^2 f_3}{\partial x \partial y})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial x \partial z})^2 + (\frac{\partial^2 f_2}{\partial x \partial z})^2 + (\frac{\partial^2 f_3}{\partial x \partial z})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2 + (\frac{\partial^2 f_3}{\partial y \partial x})^2},$
$\sqrt{(\frac{\partial^2 f_1}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2 + (\frac{\partial^2 f_3}{\partial y^2})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial y \partial z})^2 + (\frac{\partial^2 f_2}{\partial y \partial z})^2 + (\frac{\partial^2 f_3}{\partial y \partial z})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial z \partial x})^2 + (\frac{\partial^2 f_2}{\partial z \partial x})^2 + (\frac{\partial^2 f_3}{\partial z \partial x})^2},$
$\sqrt{(\frac{\partial^2 f_1}{\partial z \partial y})^2 + (\frac{\partial^2 f_2}{\partial z \partial y})^2 + (\frac{\partial^2 f_y}{\partial z \partial x})^2}, \sqrt{(\frac{\partial^2 f_1}{\partial z^2})^2 + (\frac{\partial^2 f_2}{\partial z^2})^2 + (\frac{\partial^2 f_3}{\partial z^2})^2} \Big).$

$\frac{\partial^2 f_1}{\partial x^2} = 6x,\ \frac{\partial^2 f_1}{\partial x \partial y} = 2y,\ \frac{\partial^2 f_1}{\partial x \partial z} = 0,$
$\frac{\partial^2 f_1}{\partial y \partial x} = 2y,\ \frac{\partial^2 f_1}{\partial y^2} = 2x,\ \frac{\partial^2 f_1}{\partial y \partial z} = -2z,$
$\frac{\partial^2 f_1}{\partial z \partial x} = 0,\ \frac{\partial^2 f_1}{\partial z \partial y} = -2z,\ \frac{\partial^2 f_1}{\partial z^2} = -2y,$
$\frac{\partial^2 f_2}{\partial x^2} = 2y,\ \frac{\partial^2 f_2}{\partial x \partial y} = 2x,\ \frac{\partial^2 f_2}{\partial x \partial z} = 2z,$
$\frac{\partial^2 f_2}{\partial y \partial x} = 2x,\ \frac{\partial^2 f_2}{\partial y^2} = 6y,\ \frac{\partial^2 f_2}{\partial y \partial z} = 0,$
$\frac{\partial^2 f_2}{\partial z \partial x} = 2z,\ \frac{\partial^2 f_2}{\partial z \partial y} = 0,\ \frac{\partial^2 f_2}{\partial z^2} = 2x,$
$\frac{\partial^2 f_3}{\partial x^2} = 0,\ \frac{\partial^2 f_3}{\partial x \partial y} = 0,\ \frac{\partial^2 f_3}{\partial x \partial z} = 0,$
$\frac{\partial^2 f_3}{\partial y \partial x} = 0,\ \frac{\partial^2 f_3}{\partial y^2} = 0,\ \frac{\partial^2 f_3}{\partial y \partial z} = 0,$
$\frac{\partial^2 f_3}{\partial z \partial x} = 0,\ \frac{\partial^2 f_3}{\partial z \partial y} = 0,\ \frac{\partial^2 f_3}{\partial z^2} = 60z,$

$$(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2 + (\frac{\partial^2 f_3}{\partial x^2})^2 = (6x)^2 + (2y)^2,$$

$$= \mid 36x^2 + 4y^2 \mid \leq 36|x|^2 + 4|y|^2 \leq 36(0.9)^2 + 4(0.9)^2 = 32.4,$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial x^2})^2 + (\frac{\partial^2 f_2}{\partial x^2})^2 + (\frac{\partial^2 f_3}{\partial x^2})^2} \leq 5.692.$$

$$(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2 + (\frac{\partial^2 f_3}{\partial x \partial y})^2 = (2y)^2 + (2x)^2,$$

$$= \mid 4y^2 + 4x^2 \mid \leq 4|y|^2 + 4|x|^2 \leq 4(0.9)^2 + 4(0.9)^2 = 6.48,$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial x \partial y})^2 + (\frac{\partial^2 f_2}{\partial x \partial y})^2 + (\frac{\partial^2 f_3}{\partial x \partial y})^2} \leq 2.546.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial x \partial z})^2 + (\frac{\partial^2 f_2}{\partial x \partial z})^2 + (\frac{\partial^2 f_3}{\partial x \partial z})^2} = 0.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial y \partial x})^2 + (\frac{\partial^2 f_2}{\partial y \partial x})^2 + (\frac{\partial^2 f_3}{\partial y \partial x})^2} \leq 2.546.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial y^2})^2 + (\frac{\partial^2 f_2}{\partial y^2})^2 + (\frac{\partial^2 f_3}{\partial y^2})^2} \leq 5.692.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial y \partial z})^2 + (\frac{\partial^2 f_2}{\partial y \partial z})^2 + (\frac{\partial^2 f_3}{\partial y \partial z})^2} = \sqrt{(-2z)^2} \leq \sqrt{2|z|} \leq \sqrt{2(0.9)} = 1.8.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial z \partial x})^2 + (\frac{\partial^2 f_2}{\partial z \partial x})^2 + (\frac{\partial^2 f_3}{\partial z \partial x})^2} \leq 1.8.$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial z \partial y})^2 + (\frac{\partial^2 f_2}{\partial z \partial y})^2 + (\frac{\partial^2 f_y}{\partial z \partial x})^2} \leq 1.8.$$

$$(\frac{\partial^2 f_1}{\partial z^2})^2 + (\frac{\partial^2 f_2}{\partial z^2})^2 + (\frac{\partial^2 f_3}{\partial z^2})^2 = (-2y)^2 + (2x)^2 + (60z)^2,$$

$$= \mid 4y^2 + 4x^2 + 3600z^2 \mid \leq 4|y|^2 + 4|x|^2 + 3600|z|^2,$$

$$\leq 4(0.9)^2 + 4(0.9)^2 + 3600(0.9)^2 = 2922.48,$$

$$\sqrt{(\frac{\partial^2 f_1}{\partial z^2})^2 + (\frac{\partial^2 f_2}{\partial z^2})^2 + (\frac{\partial^2 f_3}{\partial z^2})^2} \leq 54.060.$$

$$D_2 = \max_{(x,y,z)\in K} \Big(5.692,\ 2.546,\ 0,\ 1.8,\ 54.060\Big) = 54.060.$$

Bibliography

[1] C. Aggarwal and C. Reddy. *Data Clustering: Algorithms and Applications.* CRC Press, Boca Raton, 2014.

[2] J. Alam, N. Kevlahan, and O. Vasilyev. Simultaneous space-time adaptive wavelet solution of nonlinear parabolic differential equations. *J. Comput. Phys.*, 214:829–857, 2006.

[3] A. Angulo, L. Pérez Pozo, and F. Perazzo. A posteriori error estimator and an adaptive technique in meshless finite points method. *Eng. Anal. Bound. Elem.*, 33:1322–1338, 2009.

[4] R. Baier, L. Grüne, and S. Hafstein. Linear programming based Lyapunov function computation for Differential Inclusions. *Discrete Contin. Dyn. Syst. Ser. B.*, 17:33–56, 2012.

[5] M. Bardi and I. Capuzzo Dolcetta. *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman equations.* Birkhäuser, Boston, 1997.

[6] M. Berg, O. Cheong, M. Kerveld, and M. Overmars. *Computational geometry: Algorithms and Applications.* Springer-Verlag, Berlin, 2008.

[7] M. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84, 1989.

[8] M. D. Buhmann. Radial basis functions. In *Acta numerica, 2000*, volume 9 of *Acta Numer.*, pages 1–38. Cambridge Univ. Press, Cambridge, 2000.

[9] F. Camilli, L. Grüne, and F. Wirth. A generalization of Zubov's method to perturbed systems. *SIAM J. Control Optim.*, 40(2):496–515 (electronic), 2001.

[10] S. Cheng, T. Dey, and J. Shewchuk. *Delaunay Mesh Generation.* CRC Press, Boca Raton, 2013.

[11] S. Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent Fuzzy Systems*, 2(3), 1994.

[12] M. Dellnitz and O. Junge. Set oriented numerical methods for dynamical systems. In *Handbook of dynamical systems, Vol. 2*, pages 221–264. North-Holland, Amsterdam, 2002.

[13] C. Duarte and J. Oden. An *hp* adaptive method using clouds. *Comput. Methods Appl. Mech. Eng.*, 139:237–262, 1996.

[14] M. Floater and A. Iske. Multistep scattered data interpolation using compactly supported Radial Basis Functions. *J. Comput. Appl. Math.*, 73(1-2):65–78, 1996.

[15] T. Fries and T Belytschko. The extended/generalized finite element method: an overview of the method and its applications. *Int. J. Numer. Methods Eng.*, 84:253–304, 2010.

[16] Bernd Gärtner and Michael Hoffmann. Computational geometry lecture notes 1 hs 2012, 2013.

[17] P. Giesl. *Construction of Global Lyapunov Functions Using Radial Basis Functions.* Lecture Notes in Math. 1904, Springer, 2007.

[18] P. Giesl. Construction of a local and global Lyapunov function using Radial Basis Functions. *IMA J. Appl. Math.*, 73(5):782–802, 2008.

[19] P. Giesl and S. Hafstein. Revised CPA method to compute Lyapunov functions for nonlinear systems. *J. Math. Anal. Appl.*, 410:292–306, 2014.

[20] P. Giesl and S. Hafstein. Computation and verification of Lyapunov functions. *SIAM J. Applied Dynamical Systems*, 14(4):1663–1698, 2015.

[21] P. Giesl and S. Hafstein. Review on computational methods for Lyapunov functions. *Discrete and Continuous Dynamical Systems - Series B*, 20:2291–2331, 2015.

[22] P. Giesl and H. Wendland. Meshless collocation: error estimates with application to Dynamical Systems. *SIAM J. Numer. Anal.*, 45(4):1723–1741, 2007.

[23] J. Goodman and J. O'Rourke. *Handbook of Discete and Computational Geometry.* Chapman Hall/CRC, 2004.

[24] L. Grüne. An adaptive grid scheme for the discrete Hamilton-Jacobi-Bellman equation. *Numer. Math.*, 75(3):319–337, 1997.

[25] L. Grüne. *Asymptotic behavior of dynamical and control systems under perturbation and discretization*, volume 1783 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2002.

[26] S. Hafstein. A constructive converse Lyapunov theorem on exponential stability. *Discrete and Continuous Dynamical Systems - Series A*, 10(3):657–678, 2004.

[27] S. Hafstein. *An algorithm for constructing Lyapunov functions*. Monograph. Electron. J. Diff. Eqns., 2007.

[28] K. Hammouda and F Karray. A comparative study on data clustering techniques. *University of Waterloo, Ontario, Canada*, 2000.

[29] Ø. Hjelle and M. D. *Triangulations and Applications*. Springer-Verlag, Berlin, 2006.

[30] C. S. Hsu. *Cell-to-cell mapping*, volume 64 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1987.

[31] A. Iske. On the construction of kernel-based adaptive particle methods in numerical flow simulation. In *Recent developments in the numerics of nonlinear hyperbolic conservation laws*, volume 120 of *Notes Numer. Fluid Mech. Multidiscip. Des.*, pages 197–221. Springer, Heidelberg, 2013.

[32] S. Iyengar, K. Boroojeni, and N. Balakrishnan. *Mathematical Theories of Distributed Sensor Networks*. Springer, New York, 2014.

[33] L. Jameson. A wavelet-optimized, very high order adaptive grid and numerical method. *SIAM J. Sci. Comput.*, 19:1980–2013, 1998.

[34] J. Jessee, W. Fiveland, L. Howell, P. Colella, and R. Pember. An adaptive mesh refinement algorithm for the radiative transport equation. *J. Comput. Phys.*, 139:380–398, 1998.

[35] Z. Jian. *Development of Strong Form Methods with Applications in Computational Mechanics*. PhD thesis: National University of Singapore, Singapore, 2008.

[36] C. M. Kellett. Classical converse theorems in Lyapunov's second method. *Discrete Contin. Dyn. Syst. Ser. B.*, 20(8):2333–2360, 2015.

[37] R. Klein. *Concrete and Abstract Voronoi Diagrams*. Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1989.

[38] G. Kosec and B. Šarler. Adaptive meshfree method for the thermo-fluid problems with phase change. *WIT Trans. on Eng. Sciences.*, 69, 2010.

[39] G. Lee, H. Chung, and C. Choi. Adaptive crack propagation analysis with the element-free galerkin method. *Int. J. Numer. Methods Eng.*, 56:331–350, 2003.

[40] J. Leskovec, A. Rajaraman, and J. Ullman. *Mining of Massive Data Sets.* Cambridge University Press, Cambridge, 2014.

[41] G. Liu. *Mesh-Free Methods: Moving Beyond the Finite Element Method, second ed.* CRC Press., Boca Raton, 2009.

[42] G. Liu, B. Kee, and C Lu. A stabilized least-squares radial point collocation method (ls-rpcm) for adaptive analysis. *Comput. Methods Appl. Mech. Eng.*, 195:4843–4861, 2006.

[43] W. Liu and S. Jun. Multiple-scale reproducing kernel particle methods for large deformation problems. *Int. J. Numer. Methods Eng.*, 41:1339–1362, 1998.

[44] A. M. Lyapunov. The general problem of the stability of motion. *(Russian) Math. Soc. of Kharkov; English Translation, International Journal of Control*, 55:531–773, 1992.

[45] J. L. Massera. On Liapounoff's conditions of stability. *Annals of Mathematics*, 50:705–721, 1949.

[46] V. Moertini. Introduction to five data clustering algorithms. *INTEGRAL*, 7(2), 2002.

[47] N. Mohammed and P. Giesl. Grid refinement in the construction of Lyapunov functions using radial basis functions. *Discrete and Continuous Dynamical Systems - Series B*, 20:2453–2476, 2015.

[48] M. Moskowitz and F. Paliogiannis. *Functions of Several Real Variables.* World Scientific Publishing Co. Pte. Ltd., Singapore, 2011.

[49] T. Nguyen-Thoi, G. Liu, H. Nguyen-Xuan, and C. Nguyen-Tran. Adaptive analysis using the node-based smoothed finite element method (ns-fem). *Int. J. Numer. Methods Biomed. Eng.*, 27:198–218, 2011.

[50] A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Pranja, P. Seiler, and P. Parrilo. *SOSTOOLS: Sum of Squares Optimization Toolbox for MATLAB.* User's guide. Version 3.00 edition, 2013.

[51] P. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimiza*. PhD thesis: California Institute of Technology Pasadena, California, 2000.

[52] K. P. Persidskii. On a theorem of Liapunov. *C. R. (Dokl.) Acad. Sci. URSS*, 14:541–543, 1937.

[53] P. Pisharady, P. Vadakkepat, and L. Poh. *Computational Intelligence in Multi-feature Visual Pattern Recognition. Hand Posture and Face Recognition using Biological Inspired Approaches*. Springer Science+Business Media, Singapore, 2014.

[54] T. Plewa, T. Linde, and V. Weirs. *Adaptive Mesh Refinement-Theorey and Application*. Springer, 2005.

[55] F. Preparata and M. Shamos. *Computational geometry*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1985.

[56] T. Rabczuk and T. Belytschko. Adaptivity for structured meshfree particle methods in 2d and 3d. *Int. J. Numer. Methods Eng.*, 63:1559–1582, 2005.

[57] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Approx. Theory*, 18(3):548–585, 1995.

[58] R. Sibson. Development of strong form methods with applications in computational mechanics. In V. Barnett, editor, *Interpolating Multivariate data*, chapter 2. John Wiley and Sons, New York, 1981.

[59] V. Torra. *Information Fusion in Data Mining*. Springer-Verlag, Berlin, 2003.

[60] H. Wendland. Error estimates for interpolation by compactly supported Radial Basis Functions of minimal degree. *J. Approx. Theory*, 93:258–272, 1998.

[61] H. Wendland. *Scattered data approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2005.

[62] S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*. Springer-Verlag, New York., 2003.

[63] R. Yager and D. Filev. Generation of fuzzy rules by mountain clustering. *Journal of Intelligent Fuzzy Systems*, 2(3):209–219, 1993.

[64] X. Zhang, R. Ding, and Y. Li. Adaptive RPIM meshless method. In *Proceedings of the 2011 International Conference on Multimedia Technology (ICMT)*, pages 2388–2392. IEEE, 2011.